Tropical paths in vertex-colored graphs

Johanne Cohen[1], Giuseppe F. Italiano[2], Yannis Manoussakis[1], Nguyen Kim Thang[3], and Hong Phong Pham[1]

[1] LRI, University Paris-Saclay, Orsay, France.
[2] Department of Civil Engineering and Computer Science Engineering, University of Rome "Tor Vergata", Rome, Italy
[3] IBISC, University Paris-Saclay, Evry, France. Supported by ANR project OATA.

Abstract. A subgraph of a vertex-colored graph is said to be tropical whenever it contains each color of the initial graph. In this work we study the problem of finding tropical paths in vertex-colored graphs. There are two versions for this problem: the shortest tropical path problem (STPP), i.e., finding a tropical path with the minimum total weight, and the maximum tropical path problem (MTPP), i.e., finding a path with the maximum number of colors possible. We show that both versions of this problems are NP-hard for directed acyclic graphs, cactus graphs and interval graphs. Moreover, we also provide a fixed parameter algorithm for STPP in general graphs and several polynomial-time algorithms for MTPP in specific graphs, including bipartite chain graphs, threshold graphs, trees, block graphs, and proper interval graphs.

1 Introduction

In this paper we deal with vertex-colored graphs, which are useful in various situations. For instance, the Web graph may be considered as a vertex-colored graph where the color of a vertex represents the content of the corresponding page (red for mathematics, yellow for physics, etc.) [4]. Applications can also be found in bioinformatics (Multiple Sequence Alignment Pipeline or for multiple protein-protein Interaction networks) [6], or in a number of scheduling problems [13].

Given a vertex-colored graph, a *tropical subgraph* is a subgraph where each color of the initial graph appears at least once. Potentially, many graph properties, such as the domination number, the vertex cover number, independent sets, connected components, shortest paths etc. can be studied in their tropical version. This notion is close to, but somewhat different from the *colorful* concept used for paths in vertex-colored graphs [1,11,12] (recall that a colorful path in a vertex-colored graph G is a path with $\chi(G)$ vertices whose colors are different). It is also related to the concepts of *color patterns* or *colorful* used in bio-informatics [7]. Note that in a *tropical* subgraph two adjacent vertices can receive the same color. In this paper, we study tropical paths in vertex-colored graphs.

Throughout the paper, we let $G = (V, E)$ denote a simple undirected graph. Given a set of colors $\mathcal{C} = \{0, \ldots, c-1\}$, $G^c = (V, E)$ denotes a vertex-colored graph whose vertices are (not necessarily properly) colored by one of the colors in $\mathcal{C}$. Moreover, $G^c = (V, E, w)$ is a vertex-colored graph in which each edge e is associated to a real number $w(e)$, referred to as the *weight* of e. For any subgraph (or any set of vertices) H and a vertex v of G^c, we denote the number of vertices of H by $|H|$ and the set of colors of the vertices of H by $C(H)$. Moreover, we denote the color of the vertex v by $c(v)$ and denote the number of vertices of H whose colors is c by $v(H, c)$. The set of neighbors of v is denoted by $N(v)$. In this paper, we only consider *simple* paths, i.e., no vertex is visited more than once. Moreover, in accordance within the definitions above, a path P of G^c is said to be *tropical* if and only if each color of $\mathcal{C}$ appears at least once among the vertices of P. In this paper, we study the following two problems:

Shortest Tropical Path Problem (STPP). Given a weighted vertex-colored graph $G^c = (V, E, w)$ and two vertices s, t, find a tropical $s - t$ path with minimum total weight.

Maximum Tropical Path Problem (MTPP). Given a vertex-colored graph $G^c = (V, E)$, find a path with maximum number of colors.

Related work. In the special case where each vertex has a distinct color and all edge weights are equal, STPP reduces to the longest path problem. Besides, MTPP also reduces to the longest path problem whenever each vertex has a distinct color. The longest path problem has been widely studied in literature. It has been shown that for any constant $\epsilon > 0$, it is impossible to approximate the longest path in a general graph up to a factor $2^{(\log n)^{1-\epsilon}}$ unless NP is contained within quasi-polynomial deterministic time [10]. However, the longest path problem can be solved in polynomial time for several special classes of graphs, such as directed acyclic graphs (DAGs), trees, block graphs, interval biconvex graphs, etc. [15,16,9].

The tropical problems in vertex-colored graphs have been currently studying. We refer the interested reader to references [5,2,8] for other tropical problems in vertex-colored graphs, some ongoing works on dominating tropical sets, tropical connected subgraphs, tropical homomorphisms, and tropical matchings.

Contributions. In this paper, we aim to give dichotomy overviews on the complexity of STPP and MTPP. Specifically, on the hardness of STPP and MTPP, we show that both problems are NP-hard for DAGs, cactus graphs and interval graphs. This is in contrast to the longest path problem that is polynomial for those graph classes.

We subsequently design algorithms for STPP and MTPP. For STPP, we prove a property on the structure of optimal solution which is useful for the design of a fixed parameterized algorithm. Specifically, given any set of colors C, let P be a *shortest* path from vertex u to vertex v of G^c with its set of colors $C(P) = C$ and P' be a sub-path of P from vertex w to vertex t with its set of colors $C(P') \subseteq C(P)$. Then P' must be a *shortest* path from w to t of G^c with the set of colors $C(P')$. As a result, this yields a dynamic programming algorithm with complexity $O(2^c n^2)$, where c is the total number of colors in the input graph. This fixed parameter algorithm may turn out to be useful in practical applications of vertex-colored graphs where the number of colors is small.

For MTPP, we show that it can be solved in polynomial time for several classes of graphs such as trees, block graphs, proper interval graphs and in particular for bipartite chain graphs and threshold graphs, which are our main results related to MTPP. Specifically, we give two polynomial algorithms, one for bipartite chain graphs with running time $O(c \cdot M(m,n))$ and another for threshold graphs with running time $\max(O(c \cdot M(m,n)), O(n^4))$, where $M(m,n)$ is the running time of finding a maximum matching in a general graph with m edges and n vertices. (Currently, the best known running time $M(m,n) = O(\sqrt{n}m)$ [14].) The main idea behind those algorithms is to show that in bipartite chain graphs as well as in threshold graphs, the number of colors of any maximum tropical path is strongly related to the numbers of colors of any tropical matching. In particular, it is either exactly equal to the numbers of colors of any tropical matching, or it is one plus the numbers of colors of any tropical matching. This crucial property allows us to identify the set of candidate vertices for maximum tropical paths and to use efficient longest path algorithms [16,9] on these vertices to compute the corresponding maximum tropical paths.

Organization. In Section 2 and Section 3, we consider the STPP and MTPP problems respectively. Due to space constraints, in Section 2 we present only the hardness of STPP for DAGs, cactus graphs and also the fixed parameterized algorithm for this problem. In Section 3, we give the hardness result of MTPP for DAGs, cactus graphs and the polynomial algorithm for bipartite chain graphs as well as simple algorithms for trees, block graphs and proper interval graphs. Due to space limit, we refer the reader to the full paper which can be found on the authors' websites.

2 Shortest Tropical Paths

2.1 Hardness results for STPP

Theorem 1. *The shortest tropical path problem is NP-hard for DAGs, cactus graphs and interval graphs.*

Proof. The proof of this theorem follows Lemma 1 and Lemma 2. □

Lemma 1. *The shortest tropical path problem is NP-hard for DAGs and cactus graphs.*

Proof. We use a reduction from the Set Cover problem. Given an instance of the Set Cover problem in which the universe $U = \{x_1, x_2, \ldots, x_n\}$ and m sets $S = \{S_1, S_2, \ldots, S_m\}$ s.t. $S_i = \{x_{i1}, x_{i2}, \ldots, x_{i\alpha_i}\}$ and $x_{ij} \in U$ and the goal is to cover all elements of U by using the minimum number of sets of S, we construct a directed weighted vertex-colored graph $G^c = (V, E, w)$ so that a shortest tropical path in G^c will correspond to a minimum set cover for the original problem, as follows. Firstly, we create a directed path $(s = v_1, v_2, \ldots, v_{m+1} = t)$ in which the edge from $v_i \to v_{i+1}$ has weight $w(v_i, v_{i+1}) = L$. Next for each $1 \leq i \leq m$, we create another path from $v_i \to v_{i+1}$ as $v_i \to x_{i1} \to x_{i2} \to \ldots \to x_{i\alpha_i} \to v_{i+1}$. Each edge $(x_{ij} \to x_{i(j+1)})$ is assigned a positive weight $w(x_{ij}, x_{i(j+1)})$ so that $\sum_{j=1}^{\alpha_i-1} w(x_{ij}, x_{i(j+1)}) = H$. In addition, we assign $w(v_i, x_{i1}) = w(x_{i\alpha_i}, v_{i+1}) = H$. Here we denote H and L as heavy and light weights, respectively, and we assume that $H \ggg L$. Note that each vertex x_{ij} of the set S_i is an element x_k of the set U. Now we use $n+1$ colors including one color c_0 and each color c_i for each element x_i of U for $1 \leq i \leq n$. All vertices v_i are colored by the same color c_0, moreover in the case the vertex x_{ij} is x_k of U then we give x_{ij} the color c_k. Note that the constructed graph is a directed acyclic graph since it does not contain any directed cycle.

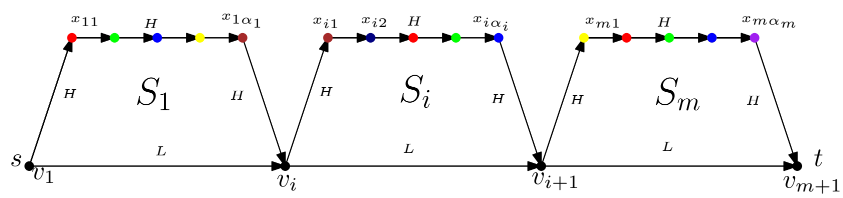

Fig. 1. Reduction of Set Cover to STPP for DAG, cactus graphs.

Now from a set cover of size t, we obtain a tropical path as follows. For each set S_i selected into this set cover, we choose the sub-path $v_i \to x_{i1} \to x_{i2} \to \ldots \to x_{i\alpha_i} \to v_{i+1}$ into our final path from s to t, otherwise the edge $v_i \to v_{i+1}$ is selected. It is clear that this path is tropical and with length $3tH + (m-t)L$. Conversely, from a tropical path of length of $3tH+(m-t)L$, we obtain a set cover of size t as follows. In the case that this tropical path uses the edge $v_i \to v_{i+1}$ then the set S_i is not selected. Otherwise, S_i is selected. It is clear that this set is a set cover since all colors are included, i.e., all elements are covered. Suppose that its size is t', then the length of the path is $3t'H + (m-t')L$. Since $3t'H + (m-t')L = 3tH + (m-t)L$, we have $(3H - L)(t - t') = 0$ and $t' = t$ since $H \ggg L$.
Thus a set cover of size t corresponds to a tropical path of length of $3tH+(m-t)L$ in G^c (and vice versa). This implies that the shortest tropical path problem is NP-hard for DAGs.
Observe that if we consider the undirected version of G^c (by ignoring the direction of edges), then our graph becomes a cactus graph, since any two simple cycles have at most one vertex in common. Thus the lemma holds also for cactus graphs. $\square$

Next we show that STPP is also NP-hard for interval graphs. The proof, is deferred to the Appendix, is an adaption from Lemma 1, with the additional idea of constructing an intersection model for our graph.

Lemma 2. *The shortest tropical path problem is NP-hard for interval graphs.*

2.2 A dynamic programming algorithm for STPP

Now we propose an algorithm for the following general problem: given a weighted vertex-colored graph $G^c = (V, E, w)$, and a fixed source $s \in V$, we wish to compute, $\forall v \in V$ and $\forall \{c_1, c_2, \ldots, c_m\} \subset \mathcal{C}$, a shortest path $p[v][2^{c_1} + 2^{c_2} + \ldots + 2^{c_m}]$ from s to v using exactly m colors $\{c_1, c_2, \ldots, c_m\}$.

Input: A weighted vertex-colored graph $G^c = (V, E, w)$, a fixed source s
Output: $\forall v \in V$ and $\forall \{c_1, c_2, \ldots, c_m\} \subset \mathcal{C}$: compute $p[v][j]$ and $d[v][j]$ as a shortest path from s to v and its length with exactly m colors $\{c_1, c_2, \ldots, c_m\}$ s.t. $j = \sum_{i=1}^{m} 2^{c_i}$, $0 \le c_1 < c_2 < \ldots < c_m \le c - 1$
Initialization: $\forall v \in V$ and $\forall 0 \le j \le 2^c$: $d[v][j] \leftarrow +\infty$; $p[v][j] \leftarrow \emptyset$;
$d[s][2^{c(s)}] \leftarrow 0$;
for $j = 0$ *to* 2^c **do**
  let $j = 2^{c_1} + 2^{c_2} + \ldots + 2^{c_m}$ s.t. $0 \le c_1 < c_2 < \ldots < c_m \le c - 1$;
  `// Step 1: initialize some values` $d[v][j]$;
  foreach $v \in V$ *s.t.* $c(v) \in \{c_1, c_2, \ldots, c_m\}$ **do**
    $j'_v \leftarrow 2^{c_1} + 2^{c_2} + \ldots + 2^{c_m} - 2^{c(v)}$;
    foreach $u \in N(v)$ *s.t.* $d[u][j'_v] < +\infty$ **do**
      if $d[v][j] > d[u][j'_v] + w(u, v)$ **then**
        $d[v][j] \leftarrow d[u][j'_v] + w(u, v)$; $p[v][j] \leftarrow p[u][j'_v] \cup \{v\}$;
      end
    end
  end
  `// Step 2: apply the core of Dijkstra's algorithm for values` $d[v][j]$;
  $B \leftarrow V \setminus \{s\}$;
  repeat
    $u \leftarrow argmin_{x \in B} d[x][j]$;
    $B \leftarrow B \setminus \{u\}$;
    foreach $v \in N(u)$ **do**
      if $d[v][j] > d[u][j] + w(u, v)$ **then**
        $d[v][j] \leftarrow d[u][j] + w(u, v)$; $p[v][j] \leftarrow p[u][j] \cup \{v\}$;
      end
    end
  until $B = \emptyset$;
end

Algorithm 1: Computing shortest paths for sets of colors

Algorithm Description. For each $0 \leq j \leq 2^c$, we let $j = 2^{c_1} + 2^{c_2} + \ldots + 2^{c_m}$ s.t. $0 \leq c_1 < c_2 < \ldots < c_m \leq c-1$: since we assume that colors are integers in $\{0, \ldots, c-1\}$, we let $d[v][j]$ denote $d[v][2^{c_1} + 2^{c_2} + \ldots + 2^{c_m}]$. The main idea behind our algorithm is to use a dynamic programming approach to compute the values $d[v][j]$. At the beginning, values $d[v][j]$ are initialized to $+\infty$. Next, suppose that the values $d[u][j']$ were correctly computed, $\forall u \in V$ and $\forall 0 \leq j' < j$. Now we show how to compute values $d[v][j]$ for $\forall v \in V$ based on values $d[u][j']$. Observe that, if there is a path from s to v with exactly m colors in $\{c_1, c_2, \ldots, c_m\}$, then the color of v (i.e., $c(v)$) must belong to the set of colors $\{c_1, c_2, \ldots, c_m\}$. Moreover, there must exist at least one vertex $u \in N(v)$ such that there is another path from s to u with all colors either in $\{c_1, c_2, \ldots, c_m\}$ or in $\{c_1, c_2, \ldots, c_m\} \setminus c(v)$. Our algorithm checks this in two steps. In the first step, we need to initialize some values $d[v][j]$ as follows. For each $v \in V$, we continuously update the value $d[v][j]$ according to paths such that each of them consists of a sub-path from s to u ($u \in N(v)$) with colors exactly in $\{c_1, c_2, \ldots, c_m\} \setminus c(v)$ and the edge (u, v). In the second step, our algorithm will consider paths from s to u ($u \in N(v)$) with colors exactly in $\{c_1, c_2, \ldots, c_m\}$ (note that those paths must contain the color $c(v)$). Thus, our algorithms updates the values $d[v][j]$ based on those two kinds of paths. This is done by using a relaxation on $d[v][j]$ for all assigned values $d[v][j]$ in the previous step, similarly to the core of Dijkstra's algorithm. The formal description is presented in Algorithm 1.

The following key lemma is useful show that Algorithm 1 correctly finds a shortest tropical path in $G^c = (V, E, w)$.

Lemma 3. *Let $v \in V$ be any vertex and let $\{c_1, c_2, \ldots, c_m\} \subset \mathcal{C}$ be any set of colors s.t. $0 \leq c_1 < c_2 < \ldots < c_m \leq c-1$. Let $j = \sum_{i=1}^{m} 2^{c_i}$. Then $p[v][j]$ is a shortest path from s to v with exactly m colors in $\{c_1, c_2, \ldots, c_m\}$, and $d[v][j]$ is the length of $p[v][j]$.*

Proof. We proceed by induction on j. We first consider the base of the induction, i.e., $j = 0$. In this case, the set of colors $\{c_1, c_2, \ldots, c_m\}$ is empty, and there is no path from s to v with an empty set of colors. Thus, $d[v][0] = +\infty$ and this value is not changed throughout the execution of our algorithm, since there does not exist any $v \in V$ such that $c(v)$ belongs to the empty set of color. Next assume that the lemma holds for $j' \leq j-1$: we show that it must also hold for j. Assume by contradiction that there exists another path $p \neq p[v][j]$ such that $w(p) < d[v][j]$ and $C(p) = \{c_1, c_2, \ldots, c_m\}$ with $j = \sum_{i=1}^{m} 2^{c_i}$. Let u be the vertex adjacent to v on p and $p' = p \setminus \{v\}$. We now distinguish two cases, depending on whether $c(v) \notin C(p')$ or $c(v) \in C(p')$. In the first case, $c(v) \notin C(p')$ and thus $C(p') \subset C(p)$. Let $j_v^{'} = 2^{c_1} + 2^{c_2} + \ldots + 2^{c_m} - 2^{c(v)} < j$ (recall that $j = \sum_{i=1}^{m} 2^{c_i}$). By the induction hypothesis, $d[u][j_v^{'}]$ is the length of a shortest path from s to u with colors in $\{c_1, c_2, \ldots, c_m\} \setminus c(v)$, and so $d[u][j_v^{'}] \leq w(p')$. According to Step 1 in our algorithm, then the final value $d[v][j]$ will satisfy that $d[v][j] \leq d[u][j_v^{'}] + w(u, v)$. This implies that $d[v][j] \leq w(p') + w(u, v) = w(p)$, which contradicts our assumption that $w(p) < d[v][j]$. In the second case, $c(v) \in$

$C(p')$ and thus $C(p') = C(p)$. Let $N_j(v) \subseteq N(v)$ be the set of neighbors of v such that for each $w \in N_j(v)$ there exists a path from s to w with all colors in $\{c_1, c_2, \ldots, c_m\}$ and v is not on this path. Note that $N_j(v) \neq \emptyset$ since $u \in N_j(v)$. Now after Step 2 of our algorithm, the value $d[v][j]$ will be smaller than or equal to the length of any path from s to v such that this path goes though a vertex in $N_j(v)$. Thus $d[v][j] < w(p)$, a contradiction. $\square$

Theorem 2. *Algorithm 1 computes the value* $d[v][\sum_{i=0}^{c-1} 2^i]$ *as the length of a shortest tropical path with all colors in* $\mathcal{C}$ *from* s *to* v *in* $O(2^c n^2)$ *time in* G^c.

Proof. The proof follows from Lemma 3: at the end of Algorithm 1, the value $d[v][\sum_{i=0}^{c-1} 2^i]$ is the length of a shortest tropical path from s to v in G^c with all colors in $\mathcal{C}$. It is easy to see that the complexity of this algorithm is dominated by the iteration for j (2^c times). Inside each iteration, we use the core of the Dijkstra's algorithm with complexity $O(n^2)$. Besides, the iteration *foreach* of v also runs $O(n^2)$ times. Therefore, the total running time of Algorithm 1 is $O(2^c n^2)$. $\square$

3 Maximum Tropical Paths

3.1 Hardness results for MTPP

As discussed above, MTPP is harder than the longest path problem. Since the longest path can not be approximated by any constant factor [10], we obtain that no polynomial-time algorithm can find a constant factor approximation for MTPP unless P=NP. We also show that MTPP is NP-hard for also in the special cases of DAGs, cactus graphs and interval graphs by using suitable reductions from MAX-SAT, as shown in the following theorem.

Theorem 3. *MTPP is NP hard for DAGs, cactus graphs and interval graphs.*

Proof. The proof follows from Lemma 4 and Lemma 5. $\square$

Lemma 4. *The maximum tropical path problem is NP-hard for DAGs and cactus graphs.*

Proof. Consider a boolean expression B in the CNF with variables $X = \{x_1, \ldots, x_s\}$ and clauses $B = \{b_1, \ldots, b_t\}$. In addition, suppose that B constains exactly 3 literals per clause (actually, we may also consider clauses of arbitrary size). We show how to construct a vertex-colored graph G^c associated with any such formula B, such that, there exists a truth assignment to the variables of B satisfying t' clauses if and only if G^c contains a path with $t' + 1$ distinct colors. Suppose that $\forall 1 \leq i \leq s$, the variable x_i appears in clauses $b_{i1}, b_{i2}, \ldots, b_{i\alpha_i}$ and $\overline{x_i}$ appears in clauses $b'_{i1}, b'_{i2}, \ldots, b'_{i\beta_i}$ in which $b_{ij} \in B$ and $b'_{ik} \in B$. Now a vertex-colored graph G^c is constructed as follows. We create $s + 1$ vertices: $s = v_1, v_2, \ldots, v_s, v_{s+1} = t$. For each vertex-pair (v_i, v_{i+1}), we create two directed paths from v_i to v_{i+1}: $(v_i \rightarrow b_{i1} \rightarrow b_{i2} \rightarrow \ldots \rightarrow b_{i\alpha_i} \rightarrow v_{i+1})$ and

$(v_i \rightarrow b'_{i1} \rightarrow b'_{i2} \rightarrow \ldots \rightarrow b'_{i\beta_i} \rightarrow v_{i+1})$. These two paths correspond to two variables x_i and $\overline{x_i}$, respectively. Now we use $t+1$ colors for G^c: a color c_0 and each color c_i for each clause b_i, $1 \leq i \leq t$. All vertices v_i are colored with c_0, $1 \leq i \leq s+1$. In the case b_{ij} is b_l in B then the vertex b_{ij} is colored with the color c_l. We proceed analogously for b'_{ik}. Note that our constructed graph is a DAG graph. Figure 2 is an illustration for our construction.

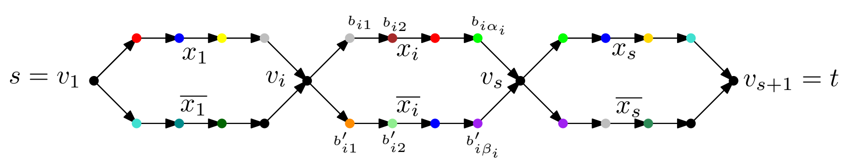

Fig. 2. Reduction of MAX-SAT problem to MTPP for DAG, cactus graphs.

Given a truth assignment for B, we obtain a path from s to t in G^c as follows. For each variable x_i which is true, we select the sub-path $(v_i \rightarrow b_{i1} \rightarrow b_{i2} \rightarrow \ldots \rightarrow b_{i\alpha_i} \rightarrow v_{i+1})$ into the final path. Otherwise, for each variable x_i which is false, we select $(v_i \rightarrow b'_{i1} \rightarrow b'_{i2} \rightarrow \ldots \rightarrow b'_{i\beta_i} \rightarrow v_{i+1})$.
Conversely, from a path from s to t in G^c, we obtain a truth assignment for B as follows. In the case our path goes though $(v_i \rightarrow b_{i1} \rightarrow b_{i2} \rightarrow \ldots \rightarrow b_{i\alpha_i} \rightarrow v_{i+1})$, then we assign x_i as true; otherwise, x_i is assigned as false. Observe that if a clause b_l is satisfied then the corresponding color c_l appears in our final path, and vice versa. Thus there exists a truth assignment to the variables of B satisfying t' clauses if and only if G^c contains a path with $t'+1$ distinct colors. In other words, $opt(G) = opt(B) + 1$ in which $Opt(G)$ is the number of colors of a maximum tropical path and $Opt(B)$ is the maximum number of satisfied clauses. As a consequence, MTPP is NP-hard for DAGs. Note that if we do not consider the directions of edges of G^c ,then we obtain a cactus graph. Thus, the lemma also holds for cactus graphs. □

We next show that MTPP is also NP-hard for interval graphs where the proof is deferred to the Appendix.

Lemma 5. *The maximum tropical path problem is NP-hard for interval graphs.*

3.2 An algorithm for MTPP in bipartite chain graphs

Recall that the longest path problem can be solved in polynomial time for bipartite permutation graphs, which can be defined as follows [16]. A bipartite permutation graph consists of bipartite chain graphs and any bipartite chain graph is a bipartite permutation graph. A bipartite graph $G = (X, Y, E)$

is said to be a *chain* graph if its vertices can be linearly ordered such that $N(x_1) \supseteq N(x_2) \supseteq \ldots \supseteq N(x_{|X|})$. As a consequence, we also have a linear order over Y such that $N(y_{|Y|}) \supseteq \ldots \supseteq N(y_1)$. It is known that these orderings over X and Y can be computed in $O(n)$ time. Here, we also use an important result in [5]: namely, that a tropical matching in vertex-colored graphs can be found in polynomial time and indeed a maximum tropical matching is also a maximum matching (in term of cardinality of the matching). The following lemma is a basic tool for our proofs.

Lemma 6. *Let M be matching in a vertex-colored bipartite chain graph $G^c = (X, Y, E)$. Then there exists a path $P(M)$ that contains all vertices of $V(M)$.*

Proof. Let $M = \{(x_{i_1}, y_{j_{|M|}}), (x_{i_2}, y_{j_{|M|-1}}), \ldots, (x_{i_{|M|}}, y_{j_1})\}$ in which $x_{i_k} \in X$ and $y_{j_k} \in Y$, $1 \leq k \leq |M|$ and $N(x_{i_1}) \supseteq N(x_{i_2}) \supseteq \ldots \supseteq N(x_{i_{|M|}})$. Now it is obvious that Since G^c is a bipartite chain graph, the edges $(x_{i_1}, y_{j_{|M|-1}})$, $\ldots$, $(x_{i_k}, y_{j_{|M|-k}})$, $\ldots$, $(x_{i_{|M|-1}}, y_{j_1})$ are in $E(G^c)$. Therefore, $P(M) = (x_{i_{|M|}}, y_{j_1}, x_{i_{|M|-1}}, y_{j_2}, \ldots, x_{i_2}, y_{j_{|M|-1}}, x_{i_1}, y_{j_{|M|}})$ is a path containing all vertices of $V(M)$. □

Now let C_m be the number of colors of any tropical matching and C_p be the number of colors of any maximum tropical path in G^c. Recall that C_m can be identified by an algorithm in [5]. The following is an important consequence of Lemma 6.

Lemma 7. *In a vertex-colored bipartite chain graph G^c, we have $C_p = C_m$ or $C_p = C_m + 1$.*

Proof. It suffices to prove that $C_m \leq C_p \leq C_m + 1$. Assume first by contradiction that $C_p < C_m$ and let M be a tropical matching with C_m colors. By Lemma 6, there exists a path P consisting of all vertices of M: clearly, $|C(P)| \geq C_m$. Thus, $|C(P)| > C_p$, a contradiction.
Assume now that $C_p > C_m + 1$, and let $P = (v_1, v_2, \ldots, v_i)$ be a maximum tropical path with C_p colors. Let $i = 2k$ if i is even, and otherwise let $i = 2k + 1$. Let $M = \{(v_1, v_2), (v_3, v_4), \ldots, (v_{2k-1}, v_{2k})\}$ be a matching in P. It is clear that $C(M) \geq C_p - 1$. Thus $C(M) > C_m$, again a contradiction. This completes our proof. □

As a consequence of Lemmas 6 and 7, the set of vertices of any maximum tropical path is either equal to the set of vertices of a tropical matching, or it differs from the the set of vertices of a tropical matching by just one vertex (see an illustration in Figure 3). In the case $C_p = C_m$, then it is possible to construct a maximum tropical path from any tropical matching based on Lemma 6. Now we consider the second case, i.e., $C_p = C_m + 1$.
Suppose that $C_p = C_m + 1$ and let P be a maximum tropical path in G^c. It is clear that the number of vertices of P is odd, i.e., $|P| = 2k + 1$. Without loss of generality, we can assume that P starts and ends with a vertex in Y, let $P = (y_{j_0}, x_{i_k}, y_{j_1}, x_{i_{k-1}}, y_{j_2}, \ldots, x_{i_2}, y_{j_{k-1}}, x_{i_1}, y_{j_k})$ in which $X' = \{x_{i_1}, \ldots, x_{i_k}\} \subseteq X$ and $Y' = \{y_{j_0}, y_{j_1}, \ldots, y_{j_k}\} \subseteq Y$. The following lemma helps to find the set X'.

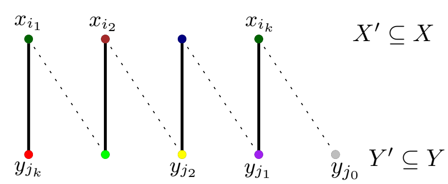

Fig. 3. An illustration for a maximum tropical path in the case $C_p = C_m + 1$.

Lemma 8. *Suppose that $C_p = C_m + 1$ and let $P = (y_{j_0}, x_{i_k}, y_{j_1}, x_{i_{k-1}}, y_{j_2}, \dots, x_{i_2}, y_{j_{k-1}}, x_{i_1}, y_{j_k})$ be a maximum tropical path of G^c. Then we have:*
(i) The set of vertices $X' = \{x_{i_1}, x_{i_2}, \dots, x_{i_k}\}$ are consecutive in the original linear ordering of X. Moreover, x_{i_1} must be x_1.
(ii) $\forall 0 \leq h \leq k$: $v(P, c(y_{j_h})) = 1$ and $|C(X')| = C_m - |X'|$.

Proof. (i): First we show that x_{i_1} must be x_1. Indeed, if $x_{i_1} \neq x_1$ then since $N(x_1) \supseteq N(x_{i_1})$, we have that $M = \{(x_1, y_{j_k}), (x_{i_1}, y_{j_{k-1}}), \dots, (x_{i_{k-1}}, y_{j_1}), (x_{i_k}, y_{j_0})\}$ is a matching such that $|C(M)| \geq |C(P)| = C_p = C_m + 1$, a contradiction.
Suppose next that the vertices $x_{i_1}, x_{i_2}, \dots, x_{i_k}$ are not consecutive in the original linear ordering of X, i.e., there exists a vertex $x_l (1 \leq l \leq |X|)$ of X ($x_l \notin X'$) and two vertices $x_{i_t}, x_{i_{t'}} \in X' (1 \leq t' \neq t \leq k)$ such that $N(x_{i_{t'}}) \supseteq N(x_l) \supseteq N(x_{i_t})$. This implies that $M = \{(x_{i_1}, y_{j_k}), (x_{i_2}, y_{j_{k-1}}), \dots, (x_{i_{t-1}}, y_{k-(t-2)}), (x_l, y_{j_{k+1-t}}), (x_{i_t}, y_{j_{k-t}}), (x_{i_{t+1}}, y_{j_{k-t-1}}), \dots, (x_{i_{k-1}}, y_{j_1}), (x_{i_k}, y_{j_0})\}$ is a matching such that $|C(M)| \geq |C(P)| = C_p = C_m + 1$, a contradiction. Thus the set of vertices X' must be consecutive in original linear ordering of X.
(ii): Now we prove that $|C(X')| = C_m - |X'|$. We claim that $\forall 0 \leq h \leq k$: $v(P, c(y_{j_h})) = 1$, i.e., the color of y_{j_h} appears only once in P. Indeed, suppose that there exists y_{j_h} s.t. $v(P, c(y_{j_h})) \geq 2$. Then, $M = \{(x_{i_1}, y_{j_k}), (x_{i_2}, y_{j_{k-1}}), \dots, (x_{i_{k-h}}, y_{j_{h+1}}), (x_{i_{k+1-h}}, y_{j_{h-1}}), (x_{i_{k+2-h}}, y_{j_{h-2}}), \dots, (x_{i_{k-1}}, y_{j_1}), (x_{i_k}, y_{j_0})\}$ is a matching in which $|C(M)| = |C(P)| = C_p = C_m + 1$, which is a contradiction. Thus $v(P, c(y_{j_h})) = 1$, $\forall y_{j_h} \in Y'$. From this property, we obtain that $|C(X')| = |C(P)| - |C(Y')| = C_m + 1 - (k + 1) = C_m - |X'|$. So we have $|C(X')| = C_m - |X'|$. □

From Lemma 8, we have that $X' = \{x_1, x_2, \dots, x_k\}$ and there is only one integer $1 \leq k \leq |X|$ which satisfies $|C(X')| = C_m - |X'|$. Thus, when $C_p = C_m + 1$ and $P = (y_{j_0}, x_{i_k}, y_{j_1}, x_{i_{k-1}}, y_{j_2}, \dots, x_{i_2}, y_{j_{k-1}}, x_{i_1}, y_{j_k})$ is a maximum tropical path of G^c, then the set X' can be found as described above. Next, we look for the set $Y' = \{y_{j_0}, y_{j_1}, \dots, y_{j_k}\} \subseteq Y$.

As proved in Lemma 8, we have that $v(P, c(y_{j_h})) = 1$, $\forall 0 \leq h \leq k$. Thus, $C(Y') \cap C(X') = \emptyset$. So to look for Y', we focus on the vertices of Y which have colors different from the colors in $C(X')$. Let $C_{Y'} = C(Y) \backslash C(X')$. Next we denote the colors of $C_{Y'}$ by $c_1, c_2, \dots, c_{|C_{Y'}|}$. Moreover for each color $c_i \in C_{Y'}$, let $max[c_i]$ be the maximum index ($1 \leq max[c_i] \leq |Y|$) such that $c(y_{max[c_i]}) = c_i$.

Moreover, without loss of generality, we can suppose that $|Y| \geq max[c_{|C_{Y'}|}] \geq \ldots \geq max[c_2] \geq max[c_1] \geq 1$. With this notation, we can reduce the search space for Y' with the help of the following lemma.

Lemma 9. *Suppose that $C_p = C_m + 1$, let $P = (y_{j_0}, x_{i_k}, y_{j_1}, x_{i_{k-1}}, y_{j_2}, \ldots, x_{i_2}, y_{j_{k-1}}, x_{i_1}, y_{j_k})$ be a maximum tropical path of G^c, and let $c_t (1 \leq t \leq |C_{Y'}|)$ be the color such that $c_t \in \{c(y_{j_0}), \ldots, c(y_{j_k})\}$ and $max[c_t] = max\{max[c(y_{j_h})] \mid 0 \leq h \leq k\}$. Then there exists another maximum tropical path P' consisting of all vertices $\{x_{i_1}, \ldots, x_{i_k}, y_{max[c_t]}, y_{max[c_{t-1}]}, \ldots, y_{max[c_{t-k}]}\}$.*

Proof. Recall that $N(y_{|Y|}) \supseteq \ldots \supseteq N(y_1)$. Now observe that since the color of y_{j_h} is $c(y_{j_h})$, we obtain that $max[c(y_{j_h})] \geq j_h, \forall 0 \leq h \leq k$. Thus $N(y_{max[c(y_{j_h})]}) \supseteq N(y_{j_h})$. As proved that the colors $c(y_{j_h})$ are distinct, $\forall 0 \leq h \leq k$. Also the colors $c(y_{max[c(y_{j_h})]})$ are distinct. Moreover, the colors $c(y_{j_h})$ and $c(y_{max[c(y_{j_h})]})$ are in $C_{Y'} = C(Y) \backslash C(X')$ and $\forall 0 \leq h \leq k$: $v(P, c(y_{j_h})) = 1$ and $v(P, c(y_{max[c(y_{j_h})]})) \leq 1$. As a result, replacing each vertex y_{j_h} in the path P by vertex $y_{max[c(y_{j_h})]}$, yields another *tropical* path P'', which is

$$(y_{max[c(y_{j_0})]}, x_{i_k}, y_{max[c(y_{j_1})]}, x_{i_{k-1}}, y_{max[c(y_{j_2})]}, \ldots, x_{i_2}, y_{max[c(y_{j_{k-1}})]}, x_{i_1}, y_{max[c(y_{j_k})]}).$$

Now since the color c_t satisfies $max[c_t] = max\{max[c(y_{j_h})] | 0 \leq h \leq k\}$ and $|Y| \geq max[c_{|C_{Y'}|}] \geq \ldots \geq max[c_2] \geq max[c_1] \geq 1$, it can be deduced that $N(y_{max[c_{t-h}]}) \supseteq N(y_{max[c(y_{j_h})]}), \forall 0 \leq h \leq k$. So in the path P'' we can replace vertices $\{y_{max[c(y_{j_0})]}, y_{max[c(y_{j_1})]}, \ldots, y_{max[c(y_{j_k})]}\}$ by vertices $\{y_{max[c_t]}, y_{max[c_{t-1}]}, \ldots, y_{max[c_{t-k}]}\}$ to obtain another *tropical* path P' consisting of all vertices $\{x_{i_1}, \ldots, x_{i_k}, y_{max[c_t]}, y_{max[c_{t-1}]}, \ldots, y_{max[c_{t-k}]}\}$. □

From Lemma 9, it follows that in order to look for Y', we must focus on $k+1$ consecutive vertices $\{y_{max[c_t]}, y_{max[c_{t-1}]}, \ldots, y_{max[c_{t-k}]}\}$ in the set of $|C_{Y'}|$ (i.e., $|C(Y) \backslash C(X')|$) vertices $\{y_{max[c_{|C_{Y'}|}]}, y_{max[c_{|C_{Y'}|-1}]}, \ldots, y_{max[c_2]}, y_{max[c_1]}\}$. It is clear that the set of $k+1$ such vertices can be easily listed. For each set $\{y_{max[c_t]}, y_{max[c_{t-1}]}, \ldots, y_{max[c_{t-k}]}\}$, together with the set $\{x_1, \ldots, x_k\}$, a path going through $2k+1$ these vertices, if it exists, can be found by an algorithm that computes *a longest path* in a bipartite chain graph [16].

When $C_p = C_m + 1$ and a maximum tropical path P starts and ends with a vertex in X, we use the notation $min[c]$ for colors in $C(X)$ (instead of $max[c]$ for colors in $C(Y)$) since the linear ordering on X is the reverse of the linear ordering on Y ($N(x_1) \supseteq N(x_2) \supseteq \ldots \supseteq N(x_{|X|})$ while $N(y_{|Y|}) \supseteq \ldots \supseteq N(y_1)$). However, in this case all other arguments go through exactly as above.

Therefore, as we find out the sets X', Y' and construct a longest path from their vertices, we check the conditions of colors to guarantee that the path has $(C_m + 1)$ colors. If it has, then it is a maximum tropical path. If we can not find such paths as all possibilities for X', Y' are considered, we conclude that a maximum tropical path must have (C_m) colors and it can be constructed from a tropical matching by Lemma 6. The formal description is presented in Algorithm 2.

Input: A vertex-colored bipartite chain graph $G^c = (X, Y, E)$ in which $N(x_1) \supseteq N(x_2) \supseteq \ldots \supseteq N(x_{|X|})$ and $N(y_{|Y|}) \supseteq \ldots \supseteq N(y_1)$

Output: A maximum tropical path with the maximum number of colors possible.

Initialization: $C_m \leftarrow$ the number of colors of a tropical matching in G^c (use the algorithm in [5]);

if $\exists k_1 (1 \leq k_1 \leq |X|)$ *such that* $|C(\{x_1, x_2, \ldots, x_{k_1}\})| = C_m - k_1$ **then**

  $X' \leftarrow \{x_1, x_2, \ldots, x_{k_1}\}$;

  $C_{Y'} \leftarrow C(Y) \backslash C(X')$;

  $\forall c \in C_{Y'}$: $max[c] \leftarrow$ the maximum index $(1 \leq max[c] \leq |Y|)$ s.t. $c(y_{max[c]}) = c$;

  $\{c_1, c_2, \ldots, c_{|C_{Y'}|}\} \leftarrow$ the set of colors of $C_{Y'}$ in which $|Y| \geq max[c_{|C_{Y'}|}] \geq \ldots \geq max[c_2] \geq max[c_1] \geq 1$;

  foreach $t \in \{k_1 + 1, \ldots, |C_{Y'}|\}$ **do**

    $Y' \leftarrow \{y_{max[c_t]}, y_{max[c_{t-1}]}, \ldots, y_{max[c_{t-k_1}]}\}$;

    $H^c \leftarrow$ the subgraph induced by vertices of $V(X')$ and $V(Y')$;

    $P \leftarrow$ the longest path of H^c (use the algorithm in [16]) ;

    if $C(P) = C_m + 1$ **then**

      return P as a maximum tropical path ;

    end

  end

else if $\exists k_2 (1 \leq k_2 \leq |Y|)$ *such that* $|C(\{y_{|Y|}, y_{|Y|-1}, \ldots, y_{k_2}\})| = C_m - k_2$ **then**

  $Y' \leftarrow \{y_{|Y|}, y_{|Y|-1}, \ldots, y_{k_2}\}$;

  $C_{X'} \leftarrow C(X) \backslash C(Y')$;

  $\forall c \in C_{X'}$: $min[c] \leftarrow$ the minimum index $(1 \leq min[c] \leq |X|)$ s.t. $c(x_{min[c]}) = c$;

  $\{c_1, c_2, \ldots, c_{|C_{X'}|}\} \leftarrow$ the set of colors of $C_{X'}$ in which $1 \leq min[c_1] \leq \ldots \leq min[c_{|C_{X'}|-1}] \leq min[c_{|C_{X'}|}] \leq |X|$;

  foreach $t \in \{1, \ldots, |C_{X'}| - k_2\}$ **do**

    $X' \leftarrow \{y_{min[c_t]}, y_{min[c_{t+1}]}, \ldots, y_{min[c_{t+k_2}]}\}$;

    $H^c \leftarrow$ the subgraph induced by vertices of $V(X')$ and $V(Y')$;

    $P \leftarrow$ the longest path of H^c (use the algorithm in [16]) ;

    if $C(P) = C_m + 1$ **then**

      return P as a maximum tropical path ;

    end

  end

else

  $M \leftarrow$ a tropical matching in G^c ;

  $P \leftarrow$ a path containing M by Lemma 6 ;

  return P as a maximum tropical path ;

end

Algorithm 2: Computing a maximum tropical path in a vertex-colored bipartite chain graph

The following theorem proves the correctness of our algorithm for computing a maximum tropical path in a vertex-colored bipartite chain graph G^c.

Theorem 4. *Algorithm 2 computes a maximum tropical path of G^c in $O(c \cdot M(m,n))$ in which $O(M(m,n))$ is the best known complexity for finding a maximum matching in a general graph with m edges and n vertices.*

Proof. The correctness of this algorithm follows from Lemma 7, Lemma 8 and Lemma 9.
This algorithm uses another algorithm to compute a tropical matching in a vertex-colored graphs [5], its complexity is $O(c \cdot M(m,n))$ in which $M(m,n)$ is the time required to compute a maximum matching in general graphs. Next the iterations **foreach** run in $O(c)$ times and inside each these iteration we use the algorithm for finding a longest path in a bipartite chain graph [16] with complexity $O(n)$. Therefore the overall complexity of Algorithm 2 is $O(c \cdot M(m,n))$. □

3.3 Algorithms for MTPP in threshold graphs

The main result for MTPP in threshold graphs is the following theorem where the proof is deferred to the Appendix.

Theorem 5. *A maximum tropical path on a threshold graph can be computed in time $max(O(c \cdot M(m,n)), O(n^4))$, where $M(m,n)$ is the time for finding a maximum matching in a general graph with m edges and n vertices.*

3.4 Algorithms for MTPP in trees, block graphs and interval graphs

In this section, we present some simple algorithms for tree, block graphs and interval graphs.

An algorithm for MTPP in trees. Observe that in a vertex-colored tree T^c, there is only a path from each vertex u to another vertex v and there are $O(n^2)$ such pairs of vertices. In this case, MTPP can be solved simply as follows:

Step 1: Compute the numbers of color of paths of all pairs of vertices (u, v) of T^c.
Step 2: Return a path with the maximum number of colors.

Algorithm 3: Look for a maximum tropical path in a vertex-colored tree T^c

An algorithm for MTPP in block graphs. As proved above, the maximum tropical path problem is NP-hard for cactus graphs (Lemma 4). However, this does not hold for other tree-like graphs, such as block graphs. We propose a polynomial algorithm for MTPP in a vertex-colored block graph G^c. Recall that a block graph is an undirected graph in which each block is a clique, it is also a clique tree. Now let u, v be two distinct vertices of $V(G^c)$, then it is clear that there exists only one series of cliques $K(u,v) = \{K_1, K_2, \ldots, K_t\}$ from u to v such that $u \in K_1$, $v \in K_t$ and K_i is adjacent to K_{i+1}, $1 \leq i \leq t-1$, moreover $K_1 \cap K_2 \neq u$ and $K_{t-1} \cap K_t \neq v$. Observe that it is possible to go through all vertices of all these cliques from u and v and it is clear that this is a longest path and also a path with maximum number of colors possible from u to v. This suggests the following simple algorithm:

Step 1: Find the longest paths between all pairs of vertices:
foreach *pair of vertices u and v in* G^c **do**
  Compute the unique series of cliques $K(u,v) = \{K_1, K_2, \ldots, K_t\}$ from u to v ;
  Find the longest path from u to v going through all vertices of $K(u,v)$, denote it by $p(u,v)$;
end
Step 2: Return a pair of vertices with the maximum number of colors of $p(u,v)$ and the corresponding path;

Algorithm 4: Computing a maximum tropical path in a vertex-colored block graph G^c

An linear algorithm for MTPP in proper interval graphs. As proved above, MTPP is NP-hard for vertex-colored interval graphs (Lemma 5). However, this problem becomes easy if we consider a vertex-colored proper interval graph G^c. Recall that proper interval graphs are interval graphs that have an interval representation in which no interval properly contains any other interval. Note that the problem of finding a longest path on proper interval graphs is easy, since all connected proper interval graphs have a Hamiltonian path which can be computed in linear time [3]. This suggests that we may compare the number of colors of Hamiltonian paths of all connected components (i.e., connected proper interval graphs) in order to select a maximum tropical path in G^c. Therefore the algorithm is simply presented as follows.

Step 1: Compute the connected components and the numbers of colors of Hamiltonian paths of all these connected components in G^c.
Step 2: Return a Hamiltonian path with the maximum number of colors.

Algorithm 5: Look for a maximum tropical path in a vertex-colored proper interval graph G^c

References

1. Akbari, S., Liaghat, V., Nikzad, A.: Colorful paths in vertex coloring of graphs. the electronic journal of combinatorics 18(1), P17 (2011)
2. Angles d'Auriac, J.A., Bujtás, C., El Maftouhi, H., Narayanan, N., Rosaz, L., Thapper, J., Tuza, Z.: Tropical dominating sets in vertex-coloured graphs. In:

Proc. 10th Workshop on Algorithms and Computation (WALCOM). vol. 9627, p. 17 (2016)
3. Bertossi, A.A.: Finding hamiltonian circuits in proper interval graphs. Information Processing Letters 17(2), 97–101 (1983)
4. Bruckner, S., Hüffner, F., Komusiewicz, C., Niedermeier, R.: Evaluation of ilp-based approaches for partitioning into colorful components. In: International Symposium on Experimental Algorithms. pp. 176–187 (2013)
5. Cohen, J., Manoussakis, Y., Pham, H., Tuza, Z.: Tropical matchings in vertex-colored graphs. In: Latin and American Algorithms, Graphs and Optimization Symposium (2017 (to appear))
6. Corel, E., Pitschi, F., Morgenstern, B.: A min-cut algorithm for the consistency problem in multiple sequence alignment. Bioinformatics 26(8), 1015–1021 (2010)
7. Fellows, M.R., Fertin, G., Hermelin, D., Vialette, S.: Upper and lower bounds for finding connected motifs in vertex-colored graphs. Journal of Computer and System Sciences 77(4), 799–811 (2011)
8. Foucaud, F., Harutyunyan, A., Hell, P., Legay, S., Manoussakis, Y., Naserasr, R.: Tropical homomorphisms in vertex-coloured graphs. Discrete Applied Mathematics ((to appear))
9. Ioannidou, K., Mertzios, G.B., Nikolopoulos, S.D.: The longest path problem is polynomial on interval graphs. In: Symposium on Symposium on Mathematical Foundations of Computer Science (MFCS). vol. 5734, pp. 403–414 (2009)
10. Karger, D., Motwani, R., Ramkumar, G.: On approximating the longest path in a graph. Algorithmica 18(1), 82–98 (1997)
11. Li, H.: A generalization of the gallai–roy theorem. Graphs and Combinatorics 17(4), 681–685 (2001)
12. Lin, C.: Simple proofs of results on paths representing all colors in proper vertex-colorings. Graphs and Combinatorics 23(2), 201–203 (2007)
13. Marx, D.: Graph colouring problems and their applications in scheduling. Periodica Polytechnica Electrical Engineering 48(1-2), 11–16 (2004)
14. Micali, S., Vazirani, V.V.: An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs. In: Proc. 21st Symposium on Foundations of Computer Science. pp. 17–27 (1980)
15. Uehara, R., Uno, Y.: Efficient algorithms for the longest path problem. In: International Symposium on Algorithms and Computation. pp. 871–883 (2004)
16. Uehara, R., Valiente, G.: Linear structure of bipartite permutation graphs and the longest path problem. Information Processing Letters 103(2), 71–77 (2007)