Pushing Vertices and Orienting Edges

William F. Klostermeyer
Department of Statistics and Computer Science
West Virginia University
Morgantown, WV 26506-6330
email: wfk@cs.wvu.edu

ABSTRACT. A directed graph operation called *pushing a vertex* is studied. When a vertex is pushed, the orientation of each of its incident edges is reversed. We consider the problems of pushing vertices so as to produce strongly connected, semi-connected, and acyclic digraphs. NP-completeness results are shown for each problem. It is shown that it is possible to create a directed path between any two vertices in a digraph; additional positive results and characterizations are shown for tournaments, outerplanar digraphs, and Hamiltonian cycles.

1 Introduction

Let $G = (V, E)$ be a directed graph (*digraph*). Define an edge orientation operation called *pushing a vertex* as follows: when a vertex is pushed, the orientation of each of its incident edges is reversed. This operation has been previously studied by Pretzel [1, 2,3] and was introduced by Mosesyan [4]. We consider the problems of deciding whether the vertices of an arbitrary digraph can be pushed so as to produce strongly connected, semi-connected, acyclic digraphs, or Hamiltonian digraphs. Each of the problems is shown to be NP-complete. The NP-completeness results imply that there exist digraphs, including infinitely many 2-edge-connected digraphs, that cannot be oriented into strongly (semi-) connected digraphs using this operation. This result contrasts the well-known theorem of Robbins [5], which states that every 2-edge connected digraph has a strong orientation if any individual edge may be oriented at will. In addition, our results show that one is unlikely to find a polynomial-time algorithm to determine if a digraph can be so oriented, let alone determine how to find a minimal set of vertices to push to produce the desired digraph. It is also shown that almost any tournament can be transformed into a strongly connected digraph using the push operation and that a broad class of digraphs which include outerplanar digraphs can be transformed into directed acyclic graphs (*dags*). The necessary conditions for transforming a digraph into one having a directed Hamiltonian cycle are also discussed.

2 Connectivity Results

Denote by (u, v) the directed edge from u to v.

2.1 NP-Completeness Results

Define the decision problem SC to be: "given a digraph G, does there exist a subset of vertices to push so that the resultant digraph is strongly connected?" We show this to be NP-complete.

Theorem 1. *SC is NP-complete.*

Proof: It is easy to see that SC is in NP, as we need only guess a subset of the vertices to push and then easily verify in polynomial time that the resultant digraph is strongly connected.

To show SC is NP-hard, we reduce SAT to SC. Recall that a SAT instance consists of a set of boolean variables, B, and a set of clauses, C, over those boolean variables [6]. C may be assumed to be in conjunctive normal form. SAT asks if there is a truth assignment that is satisfying, i.e. results in the value "true." The reduction is as follows, with an example shown in Figure 1. We assume without loss of generality that each variable or its negation appears in at least one clause, otherwise we simply ignore those variables in the reduction. For each clause $c_i \in C$, create a vertex c_i. For each boolean variable, u, create the following *variable subgraph*: vertices u, $\neg u$, u_n, u_p and directed edges (u, u_p), $(\neg u, u_p)$, (u_n, u), $(u_n, \neg u)$. Also create a special vertex T. For each *literal vertex* of the form u or $\neg u$, add the appropriate directed edge (u, T) or $(\neg u, T)$. Finally, connect each clause vertex to the variable subgraphs as follows: for each non-negated literal u in clause c_i, add directed edges $(c - i, u)$ and (c_i, u_p); for each negated literal $\neg u$ in clause c_i, add directed edges $(c_i, \neg u)$ and (c_i, u_p). The reduction clearly takes only polynomial time. Call the digraph $G = (V, E)$.

We show C is satisfiable if and only if G can be transformed into a strongly connected digraph. Suppose C is satisfiable. Then each clause c_i has a *witness*, i.e. a literal which is true. Push the associated literal vertices which are "true." That is, if non-negated literal u is assigned the value "true," push vertex u, else push vertex $\neg u$. Suppose u is the witness for c_i and is pushed. Figure 2 illustrates what happens when a "witness" vertex from Figure 1 is pushed. Clearly the subgraph induced by c_i, the variable subgraph of c_i's witness, and the special vertex T is strongly connected.

That is, each vertex in this subgraph has a directed path to T and vice versa. Since each clause has at least one witness, every vertex in G will have a directed path to T and T will have a directed path to every vertex in G. Hence G is strongly connected.

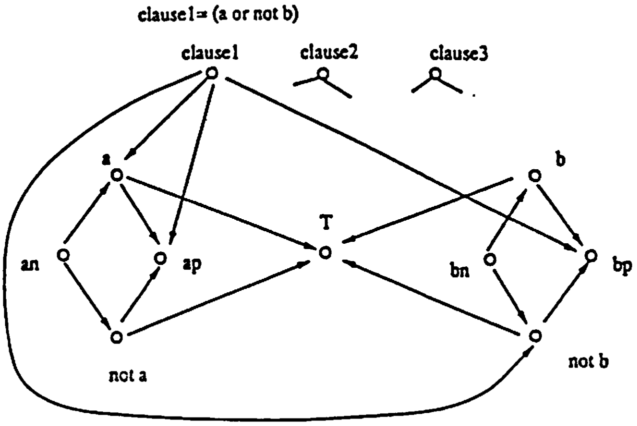

Figure 1. SAT reduction to SC

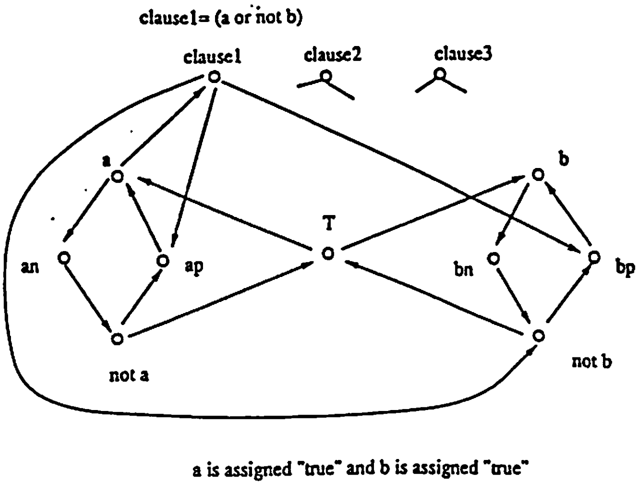

Figure 2. After a and b are pushed

Suppose G can be transformed into a strongly connected digraph, G'. It is easy to see that exactly one of each $\{u, \neg u\}$ pair must have been pushed once, since pushing both vertices makes u_n a source and then pushing u_n does not alleviate this problem (it becomes a sink). Note, the net effect of pushing a vertex twice is the same as not pushing it at all, hence we may assume each vertex is pushed either once or not at all. We derive a satisfying truth assignment as follows. If one vertex was pushed in the variable subgraph for $\{u, \neg u\}$, then assign the literal corresponding to the pushed vertex the value "true." If three vertices were pushed in the variable subgraph, assign the literal corresponding to the vertex that was not pushed the value "true." This is clearly a legal truth assignment and is also a satisfying one. This is because c_i was initially a source, hence some vertex (say, in variable subgraph U) which is adjacent to c_i must be pushed. The witness for clause c_i may be found in variable subgraph U. □

We now show that even to determine if a digraph can be transformed into a semi-connected digraph is NP-complete. Call this problem Semi-C.

Theorem 2. *Semi-C is NP-complete.*

Proof: Semi-C is easily seen to be in NP. To show NP-hardness, perform a reduction from SAT as in Theorem 1, and let G_1 be the digraph produced. Add two additional vertices to G_1, s_1 and s_2, and two additional directed edges, (T, s_1) and (s_2, T). An example is shown in Figure 3. Let G^* be the digraph that results from the reduction. It is claimed that C is satisfiable if and only if G^* can be made semi-connected. If C is satisfiable, we know from Theorem 1 that G_1 can be made into a strongly connected subgraph. After these vertices are pushed, it is easy to ensure that s_2 is a source vertex and s_1 is a sink vertex. In this way s_2 will have a directed path to every vertex in the transformed digraph, including s_1, and every vertex will have a directed path to s_1. Therefore G^* can be transformed to a semi-connected digraph.

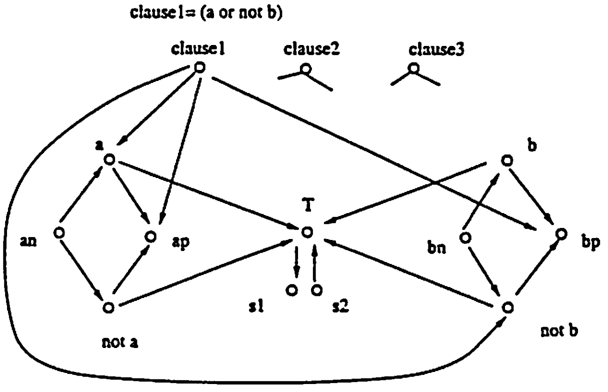

Figure 3. SAT reduction to Semi-C

On the other hand, suppose C is not satisfiable. Let G' be the digraph that results from pushing a subset of the vertices in G^* and suppose G' is semi-connected. For G' to be semi-connected, it must be that one of s_1,

s_2 is a source and the other is a sink, so that they have a directed path between them. Without loss of generality, assume s_1 is a source and s_2 is a sink. Since no vertex has a path to s_1, it must be that s_1 has a path to every vertex and therefore that T has a path to every vertex. Similarly, every vertex other than s_2 has a path to T. Therefore $G' - \{s_1, s_2\}$ is strongly connected. It now follows as in the proof of Theorem 1 that C is satisfiable. □

2.2 Finding Directed Paths in Digraphs

We show that given any pair of vertices u and v, G can be transformed so that there is a directed path from u to v, provided the underlying undirected graph is connected. The proof of this fact also gives a fast algorithm for doing so.

Fact 3. Let G be a digraph such its underlying undirected graph, G_u, is connected. Then given any pair of vertices u, v, we can transform G using the push operation so that there is a directed path from u to v.

Proof: Choose some path P between u and v in G_u. The proof is by induction on the length of P.

Base Case: $|P| = 1$. If (u, v) is a directed edge, then we have a directed path. Otherwise push v.

Inductive Hypothesis: Assume we can construct such a directed path if $|P|$ =k.

Inductive Step: $|P| = k + 1$. Using the inductive hypothesis, construct a directed path from u to w, where w is the kth vertex on the undirected path from u to v. Now simply push v, if necessary, to produce the desired directed path. □

2.3 Strong Connectivity in Tournaments

It is now shown that given almost any tournament, i.e. a digraph in which either edge (u, v) or (v, u) exists for every pair of vertices u, v, we can make the tournament strongly connected using the push operation. Observe that such a digraph is trivially semi-connected. The only tournaments that cannot be made strong are shown in Figure 4.

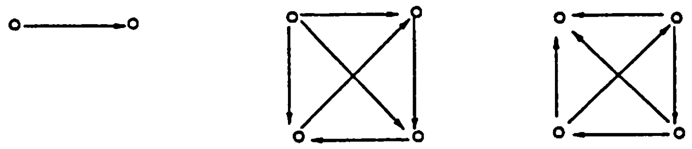

Figure 4. Tournaments that cannot be made strong

Theorem 4. *Any tournament except those in Figure 4 can be transformed into a strongly connected digraph using the push operation.*

Proof: Let $G = (V, E)$ be a tournament other than those in Figure 4. Assume for now that $|V| = 2n + 1$, i.e. $|V|$ is odd. It is well known that G contains a directed Hamiltonian path, $(v_1, v_2, \ldots, v_{2n+1})$. If (v_{2n+1}, v_1) is a directed edge in E, we are done. Otherwise (v_1, v_{2n+1}) is a directed edge, in which case we simply push vertices v_i, for $i = 2, 4, \ldots, 2n-3, 2n-1$. This reverses all the edges in the Hamiltonian path, creating a directed Hamiltonian cycle.

Now suppose $|V|$ is even. Inspection reveals that the tournaments shown in Figure 4 cannot be made strongly connected. One can also observe that any tournament on four vertices that is not isomorphic to any of the digraphs in Figure 4 can be made strongly connected. Now suppose $|V| \geq 6$ and $|V|$ is even. Partition V into two subsets, V_A and V_B, each of odd cardinality greater than or equal to three. Let G_A and G_B be the tournaments induced by this partitioning. Utilize the algorithm described in the first part of the proof to make G_A and G_B each have a directed Hamiltonian cycle. Let A be the set of vertices pushed to make G_A Hamiltonian and B the set pushed to make G_B Hamiltonian. Returning to G, push vertices in A union B, creating G'. If G' is not strongly connected, then all edges between vertices in G_A and G_B must be oriented in the same direction, say from G_A toward G_B. Let the directed Hamiltonian cycle in G_A be $(v_1, v_2, \ldots, v_k, v_1)$. There also is a directed Hamiltonian cycle in G_B, $(u_1, u_2, \ldots, u_j, u_1)$. We can make G' strongly connected by pushing the following vertices: $P = \{v_k, u_1, u_3, \ldots, u_{j-2}\}$. To see this, observe that before the vertices in P were pushed $(v_{k-1}, v_k, u_j, u_1, u_2, \ldots, u_{j-1})$ is a directed Hamiltonian path in the subgraph of G' induced by the $V(G_B)$ union $\{v_k, v_{k-1}\}$. By pushing the vertices listed above, we create a Hamiltonian cycle in that subgraph: $(v_k, v_{k-1}, u_{j-1}, u_{j-2}, \ldots, u_j, v_k)$. But we also now have a directed edge (u_1, v_1) since u_1 was pushed and v_1 was not, and we still have directed edges (v_i, v_{i+1}), for $1 \leq i \leq k-2$, and directed edge (v_{k-2}, u_2) as none of the v vertices incident to those edges were pushed. Therefore the resultant digraph is strongly connected. □

Note that Theorem 4 implies that any tournament can be made to have a directed Hamiltonian cycle, by the classic result of [7]. From the result of [8] we get the stronger result that every vertex in a strong tournament is contained in a cycle of length k, for $k = 3, 4, \ldots, |V|$.

3 Producing DAGs and Hamiltonian Digraphs

3.1 NP-Completeness of Transforming a Digraph to a DAG

It is now shown that it is NP-complete to decide if an asymmetric digraph can be transformed into a dag. By asymmetric, we mean (u, v) in E implies (v, u) not in E. If there exist vertices u and v in G such that (u, v) and (v, u) are both directed edges in G, it is obvious G cannot be transformed into a dag. Thus no symmetric digraph can be transformed to a dag.

Theorem 5. *It is NP-complete to decide if an asymmetric digraph, $G = (V, E)$, can be transformed into a directed acyclic graph (dag) using the push operation.*

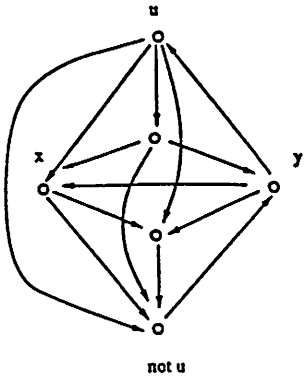

Figure 5. Variable Component

Proof: Let us denote the problem by TDAG (transform to dag). We reduce Not All Equal 3-SAT (NAE3-SAT) [6] to TDAG. A NAE3-SAT instance consists of a set of boolean variables, B, and a set of clauses, C, over those boolean variables. C may be assumed to be in conjunctive normal form. NAE3-SAT asks if there is a truth assignment such that each clause has at least one true literal and at least one false literal. For each variable u in B, create a variable component subgraph as in Figure 5. Form a cycle of length three (a clause cycle) among the three literals in each clause and call the resulting digraph G. This is clearly a polynomial time reduction. An example is shown in Figure 6. We claim there is a satisfying not-all-equal truth assignment if and only if G can be transformed into a dag. First suppose G can be transformed to a dag G^*. This direction follows from the observation that exactly one of the literal vertices $\{u, \neg u\}$ for each variable must be pushed in order to make the subgraph induced by the variable component acyclic. This leads to a valid truth assignment by associating a pushed literal vertex with a "true" literal in the boolean expression. That is, variable u is assigned "true" if and only if vertex u is pushed. Furthermore, each clause must have at least one "true" witness and one "false" witness since in the clause cycles, such as (u, v, w, u) in Figure 5, can be broken if and only if one or two of the vertices in the cycle are pushed. Now suppose C is satisfiable. Using the mapping between truth assignment and pushed literal vertices described above, it is easy to see that G can be transformed into an acyclic dag. □

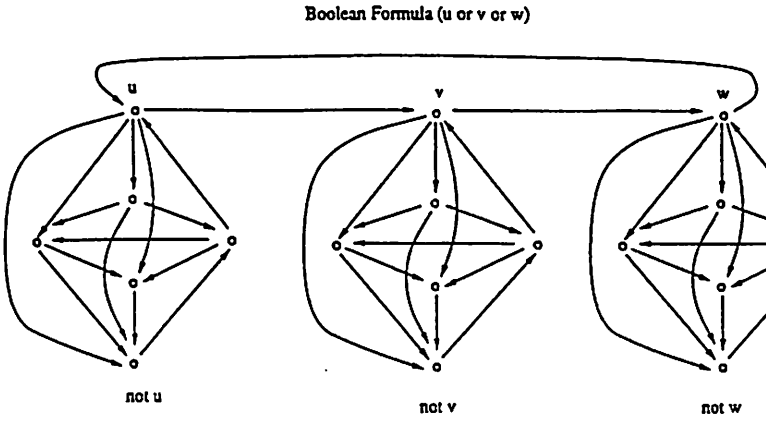

Figure 6. NAE 3-Sat Graph

3.2 Digraphs that can be Transformed to DAGs

Define a degree-two permutation digraph (D2P-graph) to be a graph such that there exists a permutation $v_1, v_2, \ldots, v_n$ of the vertices such that v_i has degree at most two in the subgraph induced by $\{v_1, v_2, \ldots, v_i\}$. Clearly any subgraph of a D2P-graph is itself a D2P-graph.

Fact 6. Any orientation of a D2P-graph $G = (V, E)$ can be transformed into a dag using the push operation.

Proof: By induction on the number of vertices.

Base Case: $|V| = 1$. Trivial.

Inductive Hypothesis: Assume true for $|V| = n$.

Inductive Step: $|V| = n + 1$. By the inductive hypothesis, assume G is acyclic prior to the addition of v_{n+1}. If the addition of v_{n+1} does not create any cycles, we are done. Suppose the addition of v_{n+1} creates one or more cycles. From the inductive hypothesis, we may assume the degree of v_{n+1} is at most two (in the underlying undirected graph). We claim that pushing v_{n+1} renders G acyclic. Suppose otherwise. Then there was a directed cycle $v_1, \ldots, v_k, v_{n+1}, v_1$ in the digraph before v_{n+1} was pushed and a directed cycle $v_1, v_{n+1}, v_k, \ldots, v_1$ in the digraph after v_{n+1} was pushed. Note that at least one of these two cycles must be of length greater than three. But since v_{n+1} was the only vertex pushed, this implies there is another cycle $v_1, \ldots, v_k, \ldots, v_1$, (of length at least three) that does not include v_{n+1}, which is a contradiction. □

A planar graph is outerplanar if it can be drawn with all nodes on the exterior boundary of the graph [9]. D2P-graphs include the class of outerplanar digraphs, as it is known (see for example [10]) that any outerplanar graph must have at least two vertices of degree at most two.

3.3 Hamiltonian Digraphs

Another logical question to ask is whether a digraph can be made to have a directed Hamiltonian path or cycle. Let us first consider the question "can G be made to have a directed Hamiltonian path" where G is an arbitrary digraph. However, in light of Fact 3, this problem is simple, if a Hamiltonian Path in the underlying, undirected digraph is specified as part of the input. If so, we can use the algorithm of Fact 3 to produce a directed Hamiltonian path in G. Otherwise, the problem requires us to determine if G has a Hamiltonian path, which is of course NP-complete [6].

Now consider Hamiltonian cycles. Suppose we are given a Hamiltonian cycle in the underlying, undirected graph as part of the input. If G has an odd number of vertices, we can use the algorithm in Theorem 4 to produce a directed Hamiltonian cycle. However, if G has an even number of vertices, we cannot always produce a Hamiltonian cycle – as may easily be seen by considering some oriented cycles on four vertices. Below, we shall characterize when a digraph with an even number of vertices and an underlying Hamiltonian cycle can be made to have a directed Hamiltonian cycle.

On the other hand, suppose we are given an arbitrary digraph and wish to determine if it can be made to have a directed Hamiltonian cycle. As one would expect, this is NP-complete.

Fact 7. Given a digraph G, determining if G can be transformed to a digraph containing a directed Hamiltonian cycle is NP-complete.

Proof: The problem is easily seen to be in NP. We prove the question to be NP-hard by reducing the Hamiltonian cycle problem in an undirected graph G_1 to our question. If G_1 has an odd number of vertices, construct a digraph G_H by providing an arbitrary orientation of the edges of G_1. From the algorithm described in Theorem 4, it is easy to see that G_H can be transformed to a digraph with a directed Hamiltonian cycle if and only if G_1 contains a Hamiltonian cycle. If G_1 has an even number of vertices, construct digraph G_H from G_1 as follows: G_H has the same vertex set as G_1; for each undirected edge (u, v) in G_1, create edges (u, v) and (v, u) in G_H. It is clear that G_1 has a Hamiltonian cycle if and only if G_H can be transformed to have a directed Hamiltonian cycle by pushing the empty set of vertices. □

We now give the condition under which a digraph with an underlying undirected Hamiltonian cycle having an even number of vertices can be transformed to a directed Hamiltonian cycle. For simplicity, we consider digraphs that are oriented n-cycles, i.e., the underlying undirected graph is connected and has n vertices and n edges. Assume a circular representation of G and number the vertices from 1 to n in a clockwise direction. Directed edges are identified as being oriented in a "clockwise" or "counterclockwise" direction.

Theorem 8. *A directed digraph G that is an oriented n-cycle with an even number of vertices can be transformed into a directed Hamiltonian cycle using the push operation if and only if G has an even number of clockwise edges.*

Proof: First suppose $G = (V, E)$ has an odd number of clockwise edges. Then G has an odd number of counterclockwise edges. Observe that pushing a vertex v in G has one of three effects: 1) no change in the number of clockwise edges (if v is a source or a sink), 2) increases the number of clockwise edges by two (if v has indegree equal to one and its incident edges were counterclockwise prior to the push), 3) decreases the number of clockwise edges by two (if v has indegree equal to one and its incident edges were clockwise prior to the push). Thus it is not possible to produce a digraph having an even number of clockwise edges.

Now suppose G has an even number of clockwise edges. Assume without loss of generality that edge (v_1, v_2) is a clockwise edge, otherwise we either have a directed (counterclockwise) Hamiltonian cycle to begin with or we may re-label the vertices. To transform G into a directed (clockwise) cycle, execute the following algorithm:

```
for $i := 2$ to $n$ loop
    if $(v_i, v_{i-1}) \in E$ then push $v_i$
                note that edges of the form $(v_i, v_{i-1})$ are
                counterclockwise by definition
```

We claim this algorithm produces a directed clockwise cycle. The proof is by induction on e, the number of counterclockwise edges in E. The base case, $e = 0$, is trivial. Now assume the algorithm works for $e = k$. We show the algorithm works for $e = k + 2$. Let v_i, $i > 1$, be the first vertex pushed by the algorithm. This implies (v_i, v_{i-1}) is a counterclockwise edge. There are two cases to consider.

Case 1) (v_{i+1}, v_i) is a counterclockwise edge. When v_i is pushed by the algorithm, the number of counterclockwise edges in the graph is reduced by two. Furthermore, all remaining counterclockwise edges must be of the form (v_j, v_{j-1}) where $j > i+1$. The claim then follows from the inductive hypothesis.

Case 2) (v_{i+1}, v_i) is not a counterclockwise edge. When v_i is pushed by the algorithm, the number of counterclockwise edges in the graph remains unchanged. However, in this case, all remaining counterclockwise edges will be of the form (v_j, v_{j-1}) where $j >= i+1$. Therefore, the algorithm will eventually reach a point where Case 1) applies and we can reduce the number of counterclockwise edges by two and invoke the inductive hypothesis. □

Acknowledgements.

The author wishes to thank the anonymous referee for their insightful comments which greatly simplified several proofs and strengthened the paper, as well as for pointing out the papers by Pretzel.

References

[1] O. Pretzel, Orientations and Edge Functions on Graphs, in *Surveys in Combinatorics*, 1991, A. D. Keedwell, Editor, London Mathematical Society Lecture Notes Series No. 166, pp, 161–185.

[2] O. Pretzel, 'On Reorienting Graphs by Pushing Down Maximal Vertices, *Order* **3** (1986), no. 2, 135–153.

[3] O. Pretzel, On Graphs that can be Oriented as Diagrams of Ordered Sets, *Order* **2** (1985), no. 1, 25–40.

[4] K.M. Mosesyan, SSR Dokl. (1972), 134–138.

[5] H.E. Robbins, A Theorem on Graphs, with an Application to a Problem in Traffic Control, *American Math. Monthly* **46** (1939), 281–283.

[6] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, W. H. Freeman, San Francisco, 1979.

[7] P. Camion, 'Chemins et Circuits Hamiltoniens des Graphes Complets, *C.R. Acad. Sci. Paris* **249** (1959), 2151–2152.

[8] J.W. Moon, 'On Subtournaments of a Tournament, *Canadian Math. Bulletin* **9** (1966), 297–301.

[9] F. Buckley and F. Harary, *Distance in Graphs*, Addison-Wesley, Redwood, California, 1990.

[10] M. Syslo, 'Characterizations of Outerplanar Graphs, *Discrete Mathematics* **26** (1979), 47–53.