J Optim Theory Appl manuscript No.
(will be inserted by the editor)

A Note on Testing Axioms of Revealed Preference

Fabrice Talla Nobibon · Bart Smeulders · Frits C.R. Spieksma

the date of receipt and acceptance should be inserted later

Abstract This Note presents an algorithm for testing the generalized axiom of revealed preference that runs in quadratic time. We show that this algorithm can be used to solve a more general problem on graphs. Furthermore, we prove a lower bound on the running time of any algorithm for testing different axioms of revealed preference.

Keywords generalized axiom of revealed preference · directed graph · strongly connected components · arc coloring · microeconomics

Mathematics Subject Classification (2000) AMS 91A26 · AMS 91B10 · AMS 05C15 · AMS 05C85

The final publication is available at Springer via http://dx.doi.org/10.1007/s10957-014-0657-9
Communicated by Christodoulos Floudas

Fabrice Talla Nobibon
ORSTAT, Faculty of Business and Economics, KU Leuven, Belgium
FedEx Express Europe, Middle East, Indian Subcontinent & Africa

Bart Smeulders (✉)
ORSTAT, Faculty of Business and Economics, KU Leuven, Belgium
Tel.: +32 16 326927
E-mail: Bart.Smeulders@kuleuven.be

Frits C.R. Spieksma
ORSTAT, Faculty of Business and Economics, KU Leuven, Belgium

1. Introduction

Modeling and analyzing household consumption behavior is a central research topic in micro-economics since its introduction by Samuelson [1] in 1938. In particular, the concept of revealed preference has attracted a lot of attention in the literature. For recent overviews we refer to Varian [2] and Smeulders et al. [3]. Here, we restrict ourselves to investigating the computational complexity of testing three well-known axioms of revealed preference: the *weak, strong* and *generalized axioms of revealed preference* (WARP, SARP and GARP). As far as we are aware, each known method for testing GARP relies on computing the transitive closure of a directed graph, and a straightforward algorithm for such an operation runs in cubic time relative to the number of observations in the dataset. The main contribution of this Note is the description of a quadratic time algorithm, based on computing strongly connected components. This algorithm solves a graph-theoretic problem, which is a generalization of GARP. This graph-theoretic problem is of some independent interest. Furthermore, we also argue that any algorithm for testing WARP, SARP or GARP will need at least a linearitmic number of operations.

This Note is organized as follows. In Section 2, we describe WARP, SARP or GARP. Next, in Section 3 we introduce an arc coloring problem that generalizes testing GARP. In Section 4, we present an quadratic time algorithm solving this problem. Finally, in Section 5, we prove the lower bound on any algorithm for testing WARP, SARP or GARP and we conclude in Section 6.

2. Notation and Definitions

Consider a unitary household acting in an economy with m goods and suppose that we have observed n (non-negative) consumption quantity bundles $q^i = \left(q^i_1, \ldots, q^i_m\right) \in \mathbb{R}^m_+$ with corresponding positive prices $p^i = \left(p^i_1, \ldots, p^i_m\right) \in \mathbb{R}^m_{++}$, for $i = 1, \ldots, n$. The component q^i_j (respectively p^i_j), for $j = 1, \ldots, m$, corresponds with the quantity of good j bought by the household (respectively the unit price of good j) at the time of observation i, $i = 1, \ldots, n$. We denote the set of observations by $S = \left\{\left(p^i, q^i\right) : \ i \in N\right\}$, where $N = \{1, \ldots, n\}$ and we refer

to S as the *dataset*. For ease of exposition, we use $i \in N$ to refer to the observation $\left(p^i, q^i\right)$ and denote the scalar product by $p^i q^j$ for $i, j \in N$.

In revealed preference theory, a unitary household is said to *directly prefer* bundle q^i over another bundle q^j if and only if q^i was chosen while q^j was affordable at prices p^i (and hence could have been chosen); this translates into $p^i q^i \geq p^i q^j$. A household is said to *prefer* q^i over q^j if and only if there exists a sequence (possibly empty) of observations r, s, $\ldots$, t such that $p^i q^i \geq p^i q^r$, $p^r q^r \geq p^r q^s$, $\ldots$, and $p^t q^t \geq p^t q^j$. These notions of preference are used in the definition of WARP, SARP and GARP [2].

Definition 2.1 A dataset S satisfies WARP if and only if, for all observations i and j, when $q^i \neq q^j$ and the household directly prefers q^i over q^j, then $p^j q^j < p^j q^i$.

Definition 2.2 A dataset S satisfies SARP if and only if, for all observations i and j, when $q^i \neq q^j$ and the household prefers q^i over q^j, then $p^j q^j < p^j q^i$.

Definition 2.3 A dataset S satisfies GARP if and only if, for all observations i and j, when the household prefers q^i over q^j, then $p^j q^j \leq p^j q^i$.

The main differences between the axioms are as follows. WARP does not test the revealed preferences for transitivity and does not allow for indifference. Both SARP and GARP do test for transitivity, and GARP, in addition, allows for indifference. As a result, datasets which satisfy SARP also satisfy WARP and GARP. Testing whether a given dataset S satisfies WARP $\left(\text{SARP, GARP}\right)$ is defined as the following *decision problem*:

Instance: A dataset S.
Question: Does S satisfy WARP? $\left(\text{SARP, GARP}\right)$

3. Literature Review and the Arc Coloring Problem

In this section, we review the current methods for testing WARP, SARP and GARP in Section 3.1, and we present an arc coloring problem in Section 3.2.

3.1 The Current State-of-the-art Concerning WARP, SARP and GARP

Varian [2] shows that the question of testing whether a given dataset S satisfies WARP, SARP or GARP can be answered in polynomial time. We now give an informal sketch of the procedures used. This sketch assumes the dataset S is such that the bundles are pairwise distinct, i.e., $q^i \neq q^j$ for each $i \neq j$. For testing WARP, we compare for each ordered pair (i, j) of observations with $q^i \neq q^j$, the quantities $p^i q^i$ and $p^i q^j$, on the one hand, and $p^j q^j$ and $p^j q^i$, on the other hand. Because we perform $O(n^2)$ comparisons in total, we infer that testing WARP can be done in $O(n^2)$ time.

For testing SARP, a directed graph G with n vertices is built where each vertex corresponds with an observation and there is an arc from vertex i to vertex j if and only if $p^i q^i \geq p^i q^j$. Next, the dataset S satisfies SARP if and only if G is *acyclic*. Observe that building the graph can be done in time $O(n^2)$, whereas checking whether G is acyclic is done in time $O(n^2)$, using, for example, a topological ordering algorithm. Thus, testing SARP can be done in $O(n^2)$ time.

The current algorithm for testing GARP is based on computing the transitive closure [2, 4] and proceeds as follows. A binary $n \times n$ matrix M is defined to summarize the direct preference relations. The entry M_{ij} is given by $M_{ij} = 1$ if $p^i q^i \geq p^i q^j$ and $M_{ij} = 0$ otherwise. Next, the transitive closure of M, denoted by MT, is computed. Based on MT, GARP is decided by checking whether there exists a pair (i, j) satisfying $MT_{ij} = 1$ and $p^j q^j > p^j q^i$. The dataset satisfies GARP if and only if such a pair (i, j) does not exist. The bottleneck of this procedure is the computation of the transitive closure of M. Varian [4] uses the $O(n^3)$-algorithm proposed by Warshall [5] and mentions the possibility of computing the transitive closure using matrix multiplication. In the literature, the best-known algorithm for matrix multiplication on general matrices runs in time $O(n^{2.376})$ [6].

3.2 An Arc Coloring Problem

We present an *arc coloring* problem that is a generalization of GARP. We denote by $G = (V, A)$ a finite directed graph with $|V|$ vertices and $|A|$ arcs. In this Note, we are only interested in

directed graphs without loops. For two distinct vertices u and $v \in V$, we denote by $u \to v$ the arc from u to v. For a given subset $A_1 \subseteq A$ of arcs we denote by $G(A_1)$ the subgraph of G defined by $G(A_1) = (V, A_1)$. A directed graph $G = (V, A)$ is *complete* if and only if for each pair of distinct vertices u and v in V we have $u \to v \in A$ or $v \to u \in A$. A sequence of vertices $[v_0, v_1, \ldots, v_\ell, v_0]$ is called a *cycle* of length $\ell + 1$ in $G = (V, A)$ if and only if $v_{i-1} \to v_i \in A$ for $i = 1, \ldots, \ell$ and $v_\ell \to v_0 \in A$. A graph is *acyclic* if it contains no cycle; otherwise it is *cyclic*. A *vertex-induced subgraph* (subsequently called *induced subgraph* in this Note) is a subset of vertices of G together with all arcs whose endpoints are both in that subset. A *strongly connected component* (SCC) of G is a maximal induced subgraph $S = (V(S), A(S))$, where for every pair of vertices u and $v \in V(S)$, there is a cycle containing both u and v. Each directed graph has a unique partition of its vertices into strongly connected components [7]. We denote by $G_{scc} = (V, A_{scc})$ the subgraph of G containing only arcs that appear in a strongly connected component of G. Given a directed graph $G = (V, A)$ and a set $\mathscr{C}$ of colors, an *arc coloring* of G is a function $f : A \to \mathscr{C}$. We consider the decision problem, called ARC COLORING, defined as follows.

ARC COLORING

Instance: A complete directed graph $G = (V, A)$, a color set $\mathscr{C} = \{c_1, c_2, \ldots, c_k\}$, an arc coloring f of G and two subsets $\mathscr{C}_1, \mathscr{C}_2 \subseteq \mathscr{C}$ satisfying $\mathscr{C} = \mathscr{C}_1 \cup \mathscr{C}_2$.
Question: Is the coloring f such that for any cycle in G in which all arcs have colors belonging to $\mathscr{C}_1$, the color of all arcs in that cycle also belongs to $\mathscr{C}_2$?

Note that $\mathscr{C}_1$ and $\mathscr{C}_2$ need neither be non empty nor be disjoint. Notice also that a Yes-answer does not imply $\mathscr{C}_1 \subseteq \mathscr{C}_2$; $\mathscr{C}_1$ may contain colors associated with arcs that do not form cycles. For different, but related problems in which an arc coloring is given and certain properties in a directed graph are studied, we refer to [8, 9].

We now argue that the problem of testing whether a given dataset S satisfies GARP is a special case of ARC COLORING. Given a dataset $S = \left\{ \left(p^i, q^i \right) : i \in N \right\}$ we build a complete directed graph $G = (V, A)$ with n vertices, where each vertex corresponds with an observation. The color set $\mathscr{C} = \{c_1, c_2, c_3\}$ has three colors and we consider the arc coloring

function f of G defined as follows: for each $u \to v \in A$

$$f(u \to v) := \begin{cases} c_1, & \text{if } p^u q^u > p^u q^v, \\ c_2, & \text{if } p^u q^u = p^u q^v, \\ c_3, & \text{if } p^u q^u < p^u q^v. \end{cases} \tag{1}$$

Observation 3.1 ARC COLORING *with f defined by* (1)*, with $\mathscr{C}_1 = \{c_1, c_2\}$ and $\mathscr{C}_2 = \{c_2, c_3\}$ is equivalent to* GARP.

This observation follows from the fact that, with these choices of $\mathscr{C}_1$ and $\mathscr{C}_2$, a graph being a yes-instance of ARC COLORING is equivalent to having the property that each cycle in G either consists solely of arcs with color c_2, or contains an arc colored with color c_3.

In the next section, we present an algorithm for solving ARC COLORING that runs in time $O(n^2)$, and we argue that this complexity is the best possible. Observation 3.1 implies that this algorithm also decides whether a given dataset S satisfies GARP in time $O(n^2)$.

4. The Algorithm

Recall that an instance of ARC COLORING is given by a complete directed graph $G = (V, A)$, a color set $\mathscr{C} = \{c_1, c_2, \ldots, c_k\}$ with k colors, an arc coloring f of G and two subsets $\mathscr{C}_1$ and $\mathscr{C}_2$ of $\mathscr{C}$ satisfying $\mathscr{C} = \mathscr{C}_1 \cup \mathscr{C}_2$. For ease of exposition, we denote by $G(\mathscr{C}_1)$ (respectively $G(\mathscr{C}_2)$) the subgraph of G consisting of arcs with colors in $\mathscr{C}_1$ and in $\mathscr{C}_2$, respectively. For two subgraphs H and F of G, we write $H \subseteq F$ to indicate that each arc in H is also present in F. We present an algorithm based on the following result.

Lemma 4.1 *An instance of* ARC COLORING *is a Yes instance if and only if* $G(\mathscr{C}_1)_{scc} \subseteq G(\mathscr{C}_2)$.

Proof $(\Rightarrow)$ Suppose that we have a Yes instance of ARC COLORING. We will show that $G(\mathscr{C}_1)_{scc} \subseteq G(\mathscr{C}_2)$. Clearly, if there is no arc in $G(\mathscr{C}_1)_{scc}$ (each SCC is a singleton) then $G(\mathscr{C}_1)_{scc} \subseteq G(\mathscr{C}_2)$. Let $u \to v$ be an arbitrary arc in $G(\mathscr{C}_1)_{scc}$. Then, there exists a unique SCC of $G(\mathscr{C}_1)$, let us say $B = (V(B), A(B))$, such that $u \to v \in A(B)$. Because B is a SCC,

there exists a sequence of arcs $v_{i-1} \to v_i \in A(B)$, for $i = 0, \ldots, \ell$, and $v_\ell \to v_0 \in A(B)$ such that $[v_0, v_1 \ldots, v_\ell, v_0]$ is a cycle in B. Because we have a Yes instance of ARC COLORING we infer that the colors of all arcs in the cycle also belong to $\mathscr{C}_2$. This completes the proof that $G(\mathscr{C}_1)_{scc} \subseteq G(\mathscr{C}_2)$.

$(\Leftarrow)$ Now suppose that $G(\mathscr{C}_1)_{scc} \subseteq G(\mathscr{C}_2)$. Consider an arbitrary cycle, $[v_0, v_1 \ldots, v_l, v_0]$ consisting of arcs using colors from $\mathscr{C}_1$. By definition of strongly connected components, all vertices and arcs of a cycle are contained within a single *SCC*. All arcs of this cycle are thus contained in $G(\mathscr{C}_1)_{scc}$ and because $G(\mathscr{C}_1)_{scc} \subseteq G(\mathscr{C}_2)$, we infer that these arcs are also contained in $G(\mathscr{C}_2)$. The color of each of these arcs is thus in $\mathscr{C}_2$. As a result, this is a Yes instance of ARC COLORING. This completes the proof of Lemma 4.1. □

We propose the following algorithm for deciding ARC COLORING. In step 1, the subgraphs are built by checking the color of all arcs in the graph G. Depending on whether this color is in $\mathscr{C}_1, \mathscr{C}_2$ or both, they are added to graphs $G(\mathscr{C}_1), G(\mathscr{C}_2)$ or both. The second step involves computing the strongly connected components of $G(\mathscr{C}_1)$. We will be basing our complexity result on Tarjan's algorithm [10], which achieves a strong worst-case bound and is relatively simple. This algorithm uses a depth-first search, sequentially labelling all vertices in a graph while following the arcs. When previously labelled nodes are encountered a cycle exists, and the algorithm works backwards towards a root node of the corresponding strongly connected component. The main difference between Algorithm 1 and the current

Algorithm 1 An algorithm for deciding ARC COLORING

1: build the subgraphs $G(\mathscr{C}_1)$ and $G(\mathscr{C}_2)$
2: compute $G(\mathscr{C}_1)_{scc}$
3: **if** every arc present in $G(\mathscr{C}_1)_{scc}$ is also in $G(\mathscr{C}_2)$ **then** return Yes
4: **else** return No

procedure (see [2]) for testing GARP stems from the fact that the former algorithm computes the strongly connected components of a directed graph whereas the latter computes the transitive closure of a matrix.

Theorem 4.1 *Algorithm 1 solves* ARC COLORING *in time* $O(|V|^2)$. *Moreover, the complexity of Algorithm 1 is best possible.*

Proof The correctness of the algorithm results from Lemma 4.1. Let us now analyze its complexity. The first step of the algorithm, building the directed graphs $G(\mathscr{C}_1)$ and $G(\mathscr{C}_2)$, can be done in time $O(|V|^2)$ because we check the color of each arc. The second step, computing $G(\mathscr{C}_1)_{scc}$, when implemented using Tarjan's algorithm [10], is completed in time $O(|V|^2)$. Finally, the last step (the **if** loop) has a running time of $O(|V|^2)$. Therefore, the overall running time of the algorithm is bounded from above by $3 \times O(|V|^2) = O(|V|^2)$.

Furthermore, note that reading the input of ARC COLORING, i.e., learning the color of each of $|V|(|V|-1)$ arcs, is a necessary step, and takes $\Omega(|V|^2)$ time. We conclude that $O(|V|^2)$ is the best possible complexity for any algorithm that solves ARC COLORING. □

Corollary 4.1 GARP *can be tested in* $O(n^2)$ *time.*

5. Lower Bounds

Of course, a valid question is whether the $O(n^2)$ algorithm for GARP (and SARP) can be sped up even further. This is not ruled out by Theorem 4.1 because GARP (and SARP) are special cases of ARC COLORING. In this section we will derive a lower bound of $\Omega(n\log n)$ on the running time of any algorithm devised for testing either WARP, SARP or GARP. This is achieved using a reduction from the *Element Distinctness* problem [11, 12].

Instance: A set $x_1, x_2, \ldots, x_m$ of m integers.
Question: Are the integers x_i, $i = 1, \ldots, m$, pairwise distinct?

Using a topological method, Yao [12] proves that any algebraic computation tree that solves the m-Element Distinctness problem has a lower bound complexity of $\Omega(m\log m)$. We next show that this lower bound is valid for WARP, SARP and GARP by arguing that an algorithm for testing these can also be used for determining whether m integers are pairwise distinct.

Theorem 5.1 *Any algorithm for testing either* WARP, SARP *or* GARP *on a dataset* S *with* n *observations has a running time bounded from below by* $\Omega(n\log n)$.

Proof Given an instance $x_1, x_2, \ldots, x_m$ of the Element Distinctness problem, we build a dataset S as follows. We assume that there are m goods. To describe the prices and the quan-

tities of all goods for each observation we make use of a 'default' price (quantity) for each good. The vector of default prices is $(x_1 - 0.1\,, x_2 - 0.1, \ldots, x_m - 0.1)$ whereas the vector of default quantities is $(0, 0, \ldots, 0)$. Note that to describe these default vectors, we need $O(m)$ operations. We consider a dataset S with m observations where an observation is identified by the index of a good. This index means that for the considered observation, that particular good, let us say j, has the price of $x_j + 0.1$ (instead of $x_j - 0.1$ as in the default vector) whereas the quantity of that good is now 1 (instead of 0 in the default quantity vector). The price (respectively the quantity) of each remaining good is exactly its default price (respectively its default quantity). Note that all the quantities in S are pairwise distinct. Also, observe that this second part of our reduction requires $O(m)$ operations because given the default price and quantity vectors, we need exactly $O(m)$ numbers to describe the dataset S. We now prove that the dataset S satisfies WARP, SARP or GARP if and only if the considered instance of the Element distinctness problem is a yes instance.

Consider the directed graph built from S (see Section 3.1) and observe that there is an arc from i to j if and only if $p^i q^i \geq p^i q^j$; that is $x_i + 0.1 \geq x_j - 0.1$, as x_i and x_j are integers, this is equivalent to $x_i \geq x_j$. As a result, if there is a cycle $s \to u \to v \ldots z \to t \to s$ then the two-cycles $s \to u \to s$, $u \to v \to u$, $\ldots$, and $t \to s \to t$ are all present. This observation implies that if the graph does not contain any two-cycle then it does not contain any cycle and vice-versa. As a result, S satisfies WARP if and only if S satisfies SARP. Another remark is that for all pair of observations i and j we have $p^i q^i = x_i + 0.1 \neq x_j - 0.1 = p^i q^j$, because x_i and x_j are integers. This means that for any pair of observations i and j we have $p^i q^i \neq p^i q^j$. These inequalities imply that S satisfies GARP if and only if S satisfies SARP. Thus, for this particular dataset S, testing SARP, WARP or GARP is equivalent. Observe now that a two-cycle $i \to j \to i$ is present in our graph if and only if $x_i = x_j$; in other words, there is a two-cycle in our graph if and only if the two elements x_i and x_j are identical. It immediately follows that the dataset S satisfies WARP, SARP and GARP if and only if the considered instance of the Element Distinctness problem is a yes-instance. This proves that any algorithm for solving WARP, SARP or GARP can be used to solve the Element Distinctness problem. Therefore, the lower

bound of $\Omega(m \log m)$ for solving the Element Distinctness problem is directly applicable to any algorithm for solving WARP, SARP or GARP.

6. Conclusions

This Note presents an $O(n^2)$ algorithm for an arc coloring problem, where n is the number of vertices in the graph. We show that this is the best possible complexity result for this problem. We further argue that testing GARP is a special case of this problem, and as a consequence we improve upon the best known complexity for algorithms for testing GARP. Furthermore, we show that the element distinctness problem can be seen as a special case of testing WARP, SARP or GARP. As it has been proven that element distinctness cannot be tested in less than $\Omega(n \log n)$ time, this provides a lower bound of the computational complexity of tests for these axioms.

Acknowledgements The authors thank the reviewers for useful comments on this paper. Frits Spieksma's research has been partially funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office and FWO Grant G.0447.10.

References

1. Samuelson, P.: The empirical implications of utility analysis. Economica **6**(4), 344–356 (1938)
2. Varian, H.: Revealed preference. Samuelsonian Economics and the 21st Century (2006)
3. Smeulders, B., Spieksma, F., Cherchye, L., De Rock, B.: Goodness of fit measures for revealed preference tests: complexity results and algorithms. ACM Transactions on Economics and Computation **2**(1), Article 3 (2014)
4. Varian, H.: The nonparametric approach to demand analysis. Economica **50**(4), 945–976 (1982)
5. Warshall, S.: A theorem on boolean matrices. Journal of the American Association of Computing Machinery **9**(1), 11–12 (1962)
6. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation **9**(3), 251–280 (1990)
7. Skiena, S.: The Algorithm Design Manual, 2 edn. Springer-Verlag New York, Inc (2008)
8. Gutin, G., Sudakov, B., Yeo, A.: Note on alternating directed cycles. Discrete Mathematics **191**, 101–107 (1998)

9. Gourvès, L., Lyra, A., Martinhon, C., Monnot, J.: Complexity of paths, trails and circuits in arc-colored digraphs. Lecture Notes in Computer Science **6108**, 222–233 (2010)
10. Tarjan, R.: Depth-first search and linear graph algorithms. SIAM Journal on Computing **1**(2), 146–160 (1972)
11. Ben-Or, M.: Lower bounds for algebraic computation trees. In: Proceedings of 15th ACM Symposium on Theory of Computing, pp. 80–86 (1983)
12. Yao, A.C.: Lower bounds for algebraic computation trees with integer inputs. SIAM Journal on Computing **20**(4), 655–668 (1991)