DOI: 10.2478/auom-2014-0012
An. Şt. Univ. Ovidius Constanţa VERSITA **Vol. 22(1),2014, 141–153**

The Path Hyperoperation

Antonios Kalampakas, Stefanos Spartalis and Alexandros Tsigkas

Abstract

A new class of hypergroupoids, deriving from binary relations, is presented, via the introduced path hyperoperation. Its properties are investigated and its connections with Graph Theory are also delineated. Moreover, we present an application of this hyperoperation to assembly line designing and management.

1 Introduction

Hypergroupoids deriving from binary relations, namely *C-hypergroupoids* where introduced by Corsini in [6], see also [16]-[19]. The correlation between hyperstructures and binary relations in general has been also investigated in [3]-[5], [7]-[9] and [11, 14]. In the present paper we further expand the ongoing research on hypergroupoids by generalizing the concept of C-hypergroupoids via the introduced *path hyperoperation*. More precisely, given a set V and a binary relation $R \subseteq V \times V$ the path hyperoperation $\diamond_R$ assigns to every pair $(x, y) \in V \times V$ the set that consists of every element of V that belongs to at least one directed path from x to y.

It is clear that this definition is a generalization of Corsini's hyperoperation and moreover, for any binary relation $R \subseteq V \times V$ and elements $x, y \in V$, their Corsini product is always a subset of $x \diamond_R y$. The connection of this notion with directed graphs is illustrated by proving that given a graph $G = (V, R)$,

Key Words: Hypergroupoids, Directed Graphs, Path Hyperoperation, Mixed Model Assembly Lines, Assembly Line Designing.
2010 Mathematics Subject Classification: Primary 05C20, 20N20; Secondary 05C40.
Received: October 2013
Revised: November 2013
Accepted: January 2014

the corresponding path hyperoperation is not partial if and only if the graph is strongly connected.

Moreover, in the sequel, we present an application to the design and management of mixed-model assembly lines. Due to the growing trend for product variability and shorter life cycle, mixed-model assembly lines are gradually replacing the traditional single model production systems and are found all over the world in many industrial environments [2]. In mixed-model assembly lines different products are jointly manufactured in intermixed product sequences on the same line. As a consequence, typically, all models are variations of the same base product and only differ in specific customizable product attributes, also referred to as options [1]. The design of such a system requires the assignment of specific tasks to workstations with respect to a given set of precedence constraints. The so obtained assembly sequence is visualized by using directed graphs, traditionally called *precedence graphs*, first developed by Salveson [15].

In this respect, the introduced path hyperoperation is employed in order to design the precedence graph of a specific product with respect to a known set of precedence relations. Moreover a similar procedure is also utilized in order to construct the *joint precedence graph* of two or more precedence graphs encompassing all their characteristics.

In Section 2, basic concepts and results on directed graphs and hypergroupoids are presented. In Section 3 we define the *path hyperoperation* which is actually a generalization of Corsini's hyperoperation. The relation of the novel notion with graph theory is also illustrated. In the last section we present a proposed application of the path hyperoperation to the design of mixed-model assembly lines.

2 Preliminaries

A partial hypergroupoid is a pair $(H, \star)$, where H is a non-empty set, and $\star$ is a hyperoperation i.e.

$$\star : H \times H \to \mathcal{P}(H), \quad (x, y) \mapsto x \star y.$$

If $A, B \in \mathcal{P}(H)$-$\{\emptyset\}$, then

$$A \star B = \bigcup_{a \in A, b \in B} a \star b.$$

We denote by $a \star B$ (respectively, $A \star b$) the hyperproduct $A \star B$ in the case that the set A (respectively, the set B) is the singleton $\{a\}$ (respectively, $\{b\}$). Moreover, $(H, \star)$ is called a hypergroupoid if $x \star y \neq \emptyset$, for all $x, y \in H$ and it is called a degenerative (respectively, total) hypergroupoid in the case that for all $x, y \in H$, $x \star y = \emptyset$ (respectively, $x \star y = H$). Given a binary relation

$R \subseteq H \times H$ the *Corsini's hyperoperation* $*_R : H \times H \to \mathcal{P}(H)$ is defined in the following way:

$$(x, y) \mapsto x *_R y = \{z \in H \mid (x, z) \in R \text{ and } (z, y) \in R\}.$$

The hyperstructure $(H, *_R)$ is called *Corsini's partial hypergroupoid associated with the binary relation* R or simply *partial C-hypergroupoid* and is denoted H_R (cf. [6, 10, 16, 17]). In the case that $x *_R y \neq \emptyset$, for all $x, y \in H$, then $(H, *_R)$ is called C-hypergroupoid. It can be easily seen that a partial C-hypergroupoid H_R is a C-hypergroupoid if and only if it holds $R \circ R = H \times H$, where $\circ$ is the usual relation composition. Let $R \subseteq H \times H$ be a binary relation on the set $H = \{x_1, x_2, \ldots, x_n\}$ then the $n \times n$ matrix

$$M_R = [m_{i,j}]_{n \times n}, \text{ with } m_{i,j} = 1 \text{ if } (x_i, x_j) \in R \text{ and } m_{i,j} = 0, \text{ else}$$

is called the boolean matrix of R.

Formally a *concrete directed graph* G is a pair (V, R_G) where:

- V is a finite set, the elements of which we call vertices and
- $R_G \subseteq V \times V$ is a set of pairs of V the elements of which we call edges.

A vertex is simply drawn as a node and an edge as an arrow connecting two vertices the head and the tail of the edge. A graph $G' = (V_{G'}, R_{G'})$ is a *subgraph* of the graph $G = (V, R_G)$ if it holds $V'_G \subseteq V$ and $R'_G \subseteq R_G$. In the other direction, a *supergraph* of a graph G is a graph that has G as a subgraph. We say that the graph H is included in the graph G ($H \leq G$) if G has a subgraph that is equal or isomorphic to H. The relation $\leq$, which is called *graph inclusion*, is a graph invariant.

Given a graph $G = (V, R_G)$ and $v \in V$ the number of edges that "leave" the vertex v is called the *out degree* of v and the number of edges that "enter" the vertex is called the *in degree* of v. Moreover we denote by $degreeout(G)$ the set of all the out degrees of G's vertices and similarly for $degreein(G)$. The *order* of a graph is the number of its vertices, i.e. $|V|$, and the size of a graph is the number of its edges, i.e. $|R_G|$. A *loop* is an edge whose head and tail is the same vertex. An edge is *multiple* if there is another edge with the same head and the same tail. A graph is called *simple* if it has no multiple edges. A vertex is called *isolated* if there is no edge connected to it. Given a graph $G = (V, R_G)$ and $v_1, v_n \in V$, we say that there exists a *path* from the node v_1 to the node v_n if there exist nodes $v_2, \ldots, v_{n-1} \in V$, such that $(v_i, v_{i+1}) \in R_G$, for all $i = 1, \ldots, n-1$. The sequence of nodes $v_1, \ldots, v_n$ is called the path from v_1 to v_n. A node v *belongs* to this path if $v = v_i$, for some $2 \leq i \leq n-1$. We set the *length* of the path $v_1, \ldots, v_n$ to be $n-1$, i.e., equal with the number of the edges that we travel to move from v_1 to v_n.

Two graphs G and H are said to be isomorphic if there exists an isomorphism f between the vertices of the two graphs that respects the edges, i.e. it holds $(x, y) \in R_G$ if and only if $(f(x), f(y)) \in R_H$. Since the specific sets V, R_G chosen to define a concrete directed graph G are actually irrelevant we don't distinguish between two isomorphic graphs. Hence the following definition of an abstract graph. The equivalence class of a concrete directed graph with respect to isomorphism is called an *abstract directed graph* or simply *graph.* A graph property is called *invariant* if it is invariant under graph isomorphisms. In what follows we consider simple graphs without isolated vertices.

3 Path Hyperoperation and Graph Theory

In this section we introduce path hypergroupoids, which are a generalization of C-hypergroupoids, and present their connection with Graph Theory.

Given a binary relation $R \subseteq V \times V$ the *path hyperoperation* $\diamond_R : V \times V \to \mathcal{P}(V)$ is defined by:

$$a \diamond_R b = \{c \in V \mid (a, c_1) \in R, \ldots, (c_i, c) \in R,\ (c, c_{i+1}) \in R, \ldots, (c_n, b) \in R, \\ c_i \in V, i \in \{1, 2, \ldots, n\}, n \in \mathbb{N}\}.$$

The so obtained hyperstructure is called *path hypergroupoid.* It is clear that for any binary relation $R \subseteq V \times V$ and elements $x, y \in V$, their Corsini product $x *_R y$ is always a subset of $x \diamond_R y$.

From the above definition and the notion of a path inside a graph we obtain

Proposition 1. *Let $G = (V, R)$ be a directed graph and $\diamond_R$ the path hyperoperation deriving from G. Then for every $x, y \in V$ it holds*

$$x \diamond_R y = \{z \in V \mid z \text{ belongs to a path from } x \text{ to } y\}.$$

It is easy to verify that for every binary relation $R \subseteq V \times V$ and elements $x, y \in V$ it holds

$$x *_R y \subseteq x \diamond_R y,$$

where $x *_R y$ is the Corsini product of x and y.

Given a graph $G = (V, R)$, the path hyperoperation $\diamond_R$ structures the set V into a (partial) hypergroupoid called the *path (partial) hypergroupoid associated with* G. If the path hyperoperation structures V into a (non-partial) hypergroupoid then the same holds for every graph G' isomorphic with G, hence this property is a graph invariant. From the definition of the path in a graph and Proposition 1 we obtain

Proposition 2. *Let $G = (V, R)$ and $v, u \in V$, then $v \diamond_R u \neq \emptyset$ if and only if there exists a path from v to u of length greater or equal to 2.*

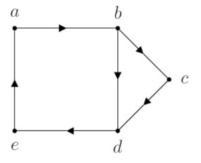
It is shown in [10] that the Corsini hyperoperation $*_R$ associated with a graph is non-partial if and only if there exists a path of length 2 between every pair of nodes of G. The path hyperoperation is a generalization of Corsini's hyperoperation and in this respect the following result is intuitively expected.

Proposition 3. *Given a graph $G = (V, R)$, the path hyperoperation $\diamond_R$ structures V into a non-partial hypergroupoid if and only if for every pair of nodes $v, u \in V$ there exists a path from v to u of length greater or equal to 2.*

Proof. The path hyperoperation associated with G is a hypergroupoid if and only if $v \diamond_G u \neq \emptyset$ for all $v, u \in V$. By Proposition 2 this holds if and only if for every pair of nodes $v, u \in V$ there exists a path from v to u of length at least 2. □

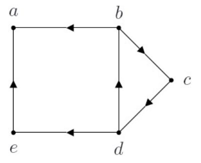
In what follows we investigate the relationship between the path hyperoperation and graph connectivity. Graph connectivity is of special interest in networking, search, shortest path and many other applications. A strongly connected graph has a path from all vertices to all vertices and the strongly connected components of a graph are the strongly connected subgraphs. The world wide web is essentially one large directed graph, to gather pages for search engines, web crawlers move from one node (page) to another along edges defined by hyperlinks. Strong connectivity can be characterized by using the path hyperoperation.

Formally, a graph $G = (V, R)$ is called *strongly connected* if for every pair of nodes $v, u \in V$ there exist paths from v to u and from u to v. For example the graph:



(G)

is strongly connected while the graph is not since from a we cannot reach



(H)

nodes b, c, d and e.

The path hyperoperation characterizes strongly connected graphs in the following way

Theorem 1. *A graph $G = (V, R)$ is strongly connected if and only if the associated path hyperoperation is non-partial.*

Proof. ($\Rightarrow$) Assume that G is strongly connected, we shall show that $v \diamond_R u \neq \emptyset$ for all $v, u \in V$. Since G is strongly connected for every $v, u \in V$ there exists a path from v to u. If the path is of length greater or equal to 2, then by Proposition 2 $v \diamond_R u \neq \emptyset$. Now assume that the path between v and u is of length 1 i.e., there exists an edge e_1 from v to u. Since we assumed that G is strongly connected there also exists a path p_1 from u to v. Hence the path $e_1 p_1 e_1$ is a path of length greater than 2 from v to u and from Proposition 2 we obtain the desired result.

($\Leftarrow$) Assume that $\diamond_R$ is non-partial i.e., for all $v, u \in V$ it holds $v \diamond_R u \neq \emptyset$. Hence from Proposition 2 there exists a path of length greater or equal to 2 from v to u. Since this holds for all pairs of nodes of G, G is strongly connected. $\square$

A graph $G' = (V', R')$ is a *subgraph* of a graph $G = (V, R)$ if $V' \subseteq V$ and $R' \subseteq R$. The strongly connected components of a graph are the strongly connected subgraphs. More formally, a *strongly connected component* of a graph G is a maximal subgraph $G' = (V', R')$ in which there exists a path between any two nodes of V'. From the above definition we obtain the following proposition.

Proposition 4. *Let $G = (V, R)$ be a graph and $v, u \in V$, then $v \diamond_R u \neq \emptyset$ and $u \diamond_R v \neq \emptyset$ if and only if v and u belong to the same strongly connected component of G.*

Proof. Similar with the proof of Theorem 1. $\square$

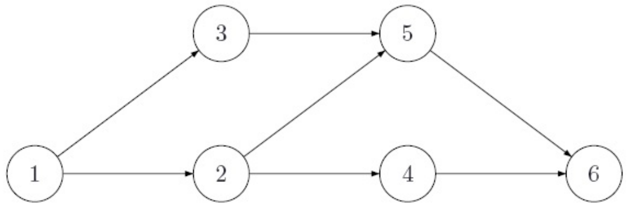
4 An Application of the Path Hyperoperation to Mixed-Model Assembly Line Designing

In this section we propose an application of the introduced hyperoperation to the design and management of mixed-model assembly lines.

The first step in designing an assembly line is the assignment of tasks to workstations. This consist of grouping the tasks in a particular way that balances the workload over a number of workstations. Traditionally an assembly sequence is visualized by virtue of *precedence graphs* first developed by Salveson [15]. The role of the precedence graph is to instruct the particular way the product has to be assembled. As a simple example, a bottle can't be filled when the cap is already on.

The fundamental problem of the design of an assembly line is to assign the tasks to an ordered sequence of workstations such that the precedence relations are satisfied in an optimal way. Many exact, heuristic and metaheuristic approaches have been proposed for solving the so-called *assembly line balancing problem*, see [12] for a review. Here we present a first step towards the employment of the theory of hyperstructures, presented in the previous sections, for the design of precedence graphs and the generation of joint precedence graphs.

More precisely, an assembly line consists of work stations usually arranged along a conveyor belt or a similar material handling equipment. The jobs are consecutively launched down the line and are moved from station to station and at each station certain operations are repeatedly performed. The total amount of work necessary to assemble a work piece (job) is split up into a set of elementary operations named tasks. Performing a task t takes a time t_j and requires certain equipment of machines and/or skills of workers. The total workload necessary for assembling a work piece is measured by the sum of task times. This arrangement can be illustrated by a precedence graph. As an example we illustrate

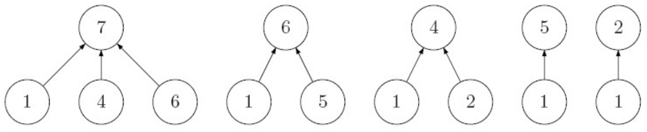


The above figure shows a precedence graph with tasks $t = 1, \ldots, 6$ (the corresponding task times are $t_1, \ldots, t_6$). It consists of a node for each task and edges for the direct precedence constraints, the indirect precedence constraints are the paths of the precedence graph.

As an application of the path hyperoperation in the assembly line designing problem we illustrate here an instance derived from the European automobile industry in the lines of [1]. First we will see how, by the use of the path hyperoperation, we can obtain the precedence graph by starting from a set of given constraints. We consider a small part of a mixed-model assembly line producing cars where the offered options are: manual sunroof, electrical sunroof, power windows and electrical mirrors. The relevant tasks $T = \{1, 2, \ldots, 7\}$ for the corresponding mixed-model assembly line are given below.

Mixed-model assembly line tasks	
1:	Initial common tasks
2:	Door wiring harness
3:	Electrical windows
4:	Electrical mirrors
5:	Roof wiring harness
6:	Sunroof installation
7:	Final common tasks

The required tasks for the model M_1 with the electrical sunroof and electrical mirrors are tasks $1, 2, 4, 5, 6$ and 7. It is known that the required precedence constraints for these tasks are as follows.

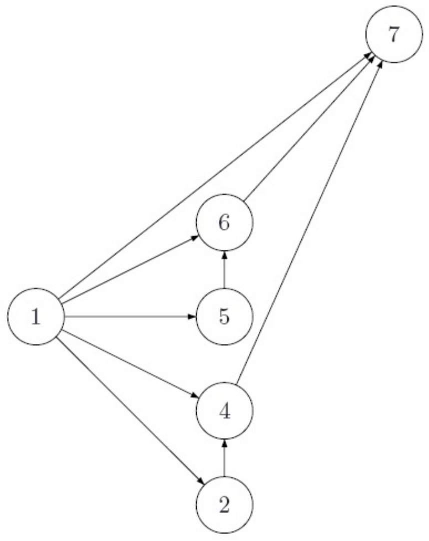


In order to design the precedence graph for the assembly of the model M_1 we first construct the corresponding binary relation

$$R_1 = \{(1,7),(4,7),(6,7),(1,6),(5,6),(1,4),(2,4),(1,5),(1,2)\}.$$

on the underlying set $V_1 = \{1, 2, 4, 5, 6, 7\}$. The related graph $G_1 = (V_1, R_1)$ is thus obtained.

G_1 :



The redundant arcs of G_1 are then removed by applying the following algorithm

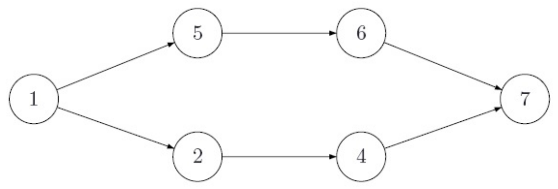
$$\text{for all } (i,j) \in R_1, \quad \text{if } i \diamond_{R_1} j \neq \emptyset \text{ then } R_1 := R_1 - (i,j) \tag{1}$$

on the table of the hyperoperation $\diamond_{R_1}$:

$\diamond_{R_1}$	1	2	4	5	6	7
1	$\emptyset$	$\emptyset$	$\{2\}$	$\emptyset$	$\{5\}$	$\{2,4,5,6\}$
2	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{4\}$
4	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
5	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{6\}$
6	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
7	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

When this procedure is completed the above graph is transformed to the desired precedence graph $G_{M_1} = (V_{M_1}, R_{M_1})$ of the model M_1.

G_{M_1} :



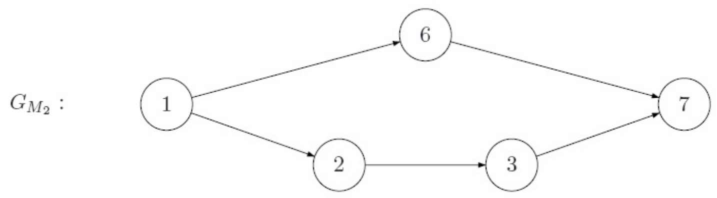
Notice that it holds $\diamond_{R_1} = \diamond_{R_{M_1}}$.

The same technique can be also utilized for the generation of the *joint* of two or more precedence graphs. Indeed car manufacturers offer their cars in a huge number of models, which can be configured by combining the options offered. The following table (see [13]) presents the variety of car types produced by European car manufacturers and the related product options (divided into four groups: car bodies, power trains, paint-and-trim-combinations and factory fitted equipment options) and the (theoretical) number of the corresponding models which grows exponentially as a function of the number of options offered.

Model	Bodies	Power trains	Paint-and-trim combinations	Factory-fitted options	Number of models
Fiat Punto	2	5	51	8	39,364
Renault Clio	2	10	57	9	81,588
Ford Fiesta	2	5	57	13	1,190,784
Renault Megane	2	6	52	14	3,451,968
GM Astra	4	11	83	14	27,088,176
GM Corsa	2	9	77	17	36,690,436
Ford Focus	4	11	64	19	366,901,933
VW Golf	3	16	221	26	1,999,813,504
Fiat Stilo	3	7	93	25	10,854,698,500
VW Polo	2	9	195	27	5.26E+10
Mini (BMW)	1	5	418	44	5.10E+16
BMW 3-Series	3	18	280	45	6.41E+16
Mercedes C-Class	2	16	312	59	1.13E+21
Mercedes E-Class	2	15	285	70	3.35E+24

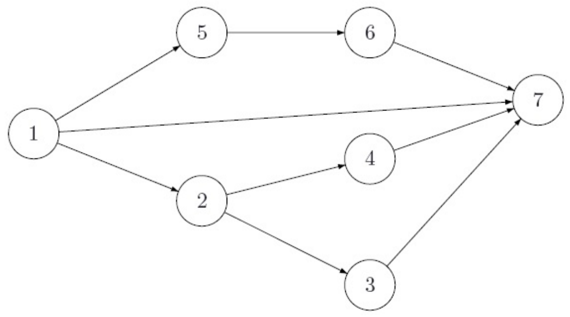
Table 1: Number of options and models for selected European cars

Actually, only a small selection of these theoretical option combinations are demanded and, thus, only very few models are repeatedly assembled but there is no adequate basis for estimating future demand rates. Moreover, a procedure for generating a joint precedence graph, which has to iterate through all possible models, suffers from the extraordinary computational requirements in this order of magnitude. Consequently, the effective generation of joint precedence graphs is of decisive importance. In what follows we illustrate the use the algorithm (1) in order to create the joint precedence graph obtained by combining two given precedence graphs. Consider the model M_1 presented above and a model M_2 with electrical windows and manual sunroof. The corresponding precedence graph $G_{M_2} = (V_2, R_2)$ is



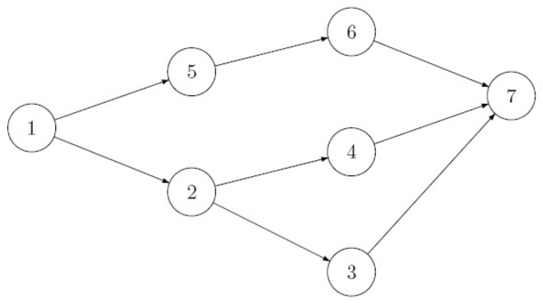
The joint precedence graph of G_{M_1} and G_{M_2} is obtained as follows. First we construct the intermediate graph $G' = (V', R')$ where $V' = V_1 \cup V_2$ and $R' = R_1 \cup R_2$. Hence the resulting graph G' is

G' :



We then apply algorithm (1) to the corresponding hyperoperation $\diamond_{R'}$ and obtain the resulting precedence graph $G_{M_1 \vee M_2}$

$G_{M_1 \vee M_2}$:



We note that although the examples presented in this section derive from the automobile industry, the proposed approach can be applied for task assignment of any mixed-model assembly line with a high degree of product variety.

Acknowledgments The publication of this paper is partially supported by the grant PN-II-IDWE-2012-4-169.

References

[1] N. Boysen, M. Fliedner, A. Scholl, Assembly line balancing: Joint precedence graphs under high product variety, IIE Transactions 41, 183-193, 2009.

[2] J. Bukchin, E. M. Dar-El, J. Rubinovitz, Mixed model assembly line design in a make-to-order environment, Computers and Industrial Engineering 41, 405-421, 2001.

[3] I. Cristea, M. Stefanescu, Binary relations and reduced hypergroups, Discrete Mathematics 308, 3537-3544, 2008.

[4] I. Cristea, M. Stefanescu, Hypergroups and n-ary relations, European Journal of Combinatorics 31, 780-789, 2010.

[5] I. Cristea, M. Stefanescu, C. Anghluta, About the fundamental relations defined on the hypergroupoids associated with binary relations, European Journal of Combinatorics 32, 72-81, 2011.

[6] P. Corsini, Binary relations and hypergroupoids, Italian Journal of Pure and Applied Mathematics 7, 11-18, 2000.

[7] P. Corsini, V. Leoreanu, Hypergroups and binary relations, Algebra Universalis 43, 321-330, 2000.

[8] P. Corsini, V. Leoreanu, Applications of Hyperstructure Theory, Kluwer Academic Publishers, Boston, Dordrecht, London, 2002.

[9] P. Corsini, V. Leoreanu, Survey on new topics of hyperstructure theory and its applications, Proc. of 8th Internat. Congress on AHA, 1-37, 2003.

[10] A. Kalampakas, S. Spartalis and K. Skoulariki, Directed Graphs representing isomorphism classes of **C**-Hypergroupoids, Ratio Mathematica 23, 51-64, 2012.

[11] M. Konstantinidou, K. Serafimidis, Sur les P-supertrillis, New Frontiers in Hyperstructures and Rel. Algebras, Hadronic Press, Palm Harbor U.S.A., 139-148, 1996.

[12] N. Kriengkorakot, N. Pianthong, The Assembly Line Balancing Problem : Review articles, KKU Engineering Journal 34, 133-140, 2007.

[13] F.K. Pil, M. Holweg, Linking product variety to order-fulfillment strategies, Interfaces 34, 394-403, 2004.

[14] I. Rosenberg, Hypergroups and Join Spaces determined by Relations, Italian Journal of Pure and Applied Mathematics 4, 93-101, 1998.

[15] M. E. Salveson, The Assembly Line Balancing Problem, Journal of Industrial Engineering 6, 18-25, 1955.

[16] S. Spartalis, Hypergroupoids obtained from groupoids with binary relations, Italian Journal of Pure and Applied Mathematics 16, 201-210, 2004.

[17] S. Spartalis, The hyperoperation relation and the Corsini's partial or not-partial hypergroupoids (A classification), Italian Journal of Pure and Applied Mathematics 24, 97-112, 2008.

[18] S. Spartalis, M. Konstantinidou, A. Taouktsoglou C-hypergroupoids obtained by special binary relations, Computers and Mathematics with Applications 59, 2628-2635, 2010.

[19] S. Spartalis, C. Mamaloukas, On hyperstructures associated with binary relations, Computers and Mathematics with Applications 51, 41-50, 2006.

Antonios Kalampakas,
College of Engineering and Technology,
American University of the Middle East, Egaila, Kuwait, and
Department of Production Engineering and Management
Laboratory of Computational Mathematics
School of Engineering, Democritus University of Thrace
V.Sofias 12, Prokat, Building A1, 67100 Xanthi, Greece
Email: akalampakas@gmail.com

Stefanos Spartalis,
Department of Production Engineering and Management
Laboratory of Computational Mathematics
School of Engineering, Democritus University of Thrace
V.Sofias 12, Prokat, Building A1, 67100 Xanthi, Greece
Email: sspart@pme.duth.gr

Alexandros Tsigkas,
Department of Production Engineering and Management
Laboratory of Computational Mathematics
School of Engineering, Democritus University of Thrace
V.Sofias 12, Prokat, Building A1, 67100 Xanthi, Greece
Email: atsigkas@pme.duth.gr