Strategic Management Journal
Strat. Mgmt. J., **37**: 177–191 (2016)
Published online EarlyView 15 December 2015 in Wiley Online Library (wileyonlinelibrary.com) DOI: 10.1002/smj.2455
Received 29 August 2013; *Final revision received 31 March 2015*

GROUPING INTERDEPENDENT TASKS: USING SPECTRAL GRAPH PARTITIONING TO STUDY COMPLEX SYSTEMS

CHI-HYON LEE,[1] MANUELA N. HOEHN-WEISS,[2]* and SAMINA KARIM[3]
[1] *Management, School of Business, George Mason University, Fairfax, Virginia, U.S.A.*
[2] *Strategy and Entrepreneurship, College of Business, Oregon State University, Corvallis, Oregon, U.S.A.*
[3] *Entrepreneurship and Innovation, D'Amore-McKim School of Business, Northeastern University, Boston, Massachusetts, U.S.A.*

Research summary: *This article uses spectral graph partitioning to advance strategic management research, and focuses on the study of complex systems that contain strongly connected components with component interactions that are weighted and directed. The spectral graph partitioning method complements existing methods, especially, when external architectural artifacts do not exist or are less than certain. We illustrate this methodology using a U.S. airline's production system. We highlight some useful metrics and show how researchers can apply this method to generate additional architectural insights.*

Managerial summary: *We describe a method for analyzing the architecture of complex systems that contain directed and weighted component interactions, and in which each component is interdependent (directly or indirectly) on every other component. Using the spectra of a complex system, the method is not only tractable but also provides good and predictable groupings. We illustrate this method using the firm-level production task system of a firm found in the U.S. passenger airline sector. The method is useful if the system architecture is hidden, in flux, or both. The method may also permit a holistic comparison of different systems and their architectures. We discuss metrics and illustrate how they can provide additional architectural insights.*

INTRODUCTION

A key issue in strategic management is how to manage complex organizational and technological systems that enable firms to compete effectively in dynamic environments (Eisenhardt and Martin, 2000; Galunic and Eisenhardt, 2001; Pil and Cohen, 2006). To better understand this issue, a growing body of literature within management uses a conceptual anchor of architectures (including technical product design, production, and organizational) as complex systems. Simon (1962) and Alexander (1964) articulated general system structuring principles that many researchers consider the beginning of a "general theory" of complex systems. Simon (1962) defined a *complex system* as "one made up of a large number of parts that interact in a nonsimple way. In such systems, the whole is more than the sum of the parts … [thus] it is not a trivial matter to infer properties of the whole" (p. 468).

In a stylized fashion, scholars have used this conceptual anchor and built on the works of Simon

Keywords: architecture; modularity; interdependence; complex systems; graph partitioning
*Correspondence to: Manuela N. Hoehn-Weiss, Strategy and Entrepreneurship, College of Business, Oregon State University, 443 Austin Hall, Corvallis, OR 97331, U.S.A. E-mail: Manuela.HW@oregonstate.edu

(1962) and Alexander (1964) in one of two ways. One stream emphasizes architectural *properties* of complex systems, such as system connectivity (e.g., Ethiraj and Levinthal, 2004; Rivkin and Siggelkow, 2007), the analysis of local structures (Karim, 2006; Lomi and Pattison, 2006; Mihm, Loch, and Huchzermeier, 2003; Schilling and Phelps, 2007), and their impact on system performance. In a complementary but distinct body of research, scholars implicitly or explicitly examine system architecture at *three* distinct levels (i.e., system, module or subsystem, and component) because one "can distinguish between the interactions *among* subsystems, on the one hand, and the interactions *within* subsystems—that is, among the parts of those subsystems—on the other" (Simon, 1962: 473; emphasis in original). For a recent extant literature review, see Campagnolo and Camuffo (2010). Researchers in this stream have examined whether an architecture is modular, integral, or somewhere in between these extremes (Baldwin and Clark, 2000; Ulrich, 1995) and its effect on various measures of system performance. The effect of the system-level architecture on firm outcomes such as (1) firm structure and its boundaries (e.g., Hoetker, 2006; Hoetker, Swaminathan, and Mitchell, 2007; Ulrich and Ellison, 1999), (2) industry structure (e.g., Fixson and Park, 2008; Staudenmayer, Tripsas, and Tucci, 2005), and (3) performance (e.g., Baldwin and Clark, 2000; Pil and Cohen, 2006; Worren, Moore, and Cardona, 2002) are some examples of these studies. Another growing body of research in this stream also examines the product architecture and firm structure alignment and subsequent impact on firm outcomes (e.g., Furlan, Cabigiosu, and Camuffo, 2014; Gokpinar, Hopp, and Iravani, 2010; Sanchez and Mahoney, 1996; Sosa, Eppinger, and Rowles, 2004; Sosa, Mihm, and Browning, 2013). A final large body of extant research within the stream examining architecture at three distinct levels has evaluated different modularity metrics (e.g., Chiriac *et al.*, 2011; Holtta-Otto *et al.*, 2012).

We contribute to this body of work by introducing to management research a spectral graph partitioning methodology that enables the examination of complex systems possessing three specific characteristics that in the past, to the best of our knowledge, could not be analyzed at the three aforementioned levels simultaneously. Management studies of product, production, or organizational structure modularity often discuss the components within these subsystems as *tasks* that are to varying degrees independent or interdependent. First, system component and module interactions may be directed (e.g., $A \rightarrow B$ but $B \nrightarrow A$). Second, system component and module interactions may be weighted (positive real numbers); these interactions or dependencies (between modules as well as between components within a module) also vary in terms of frequency, intensity, or both (Simon, 1962). Thus, task interactions may be unequal in magnitude. Third, system components and modules may be strongly connected—namely, *every* component and module directly or indirectly is interdependent with *every other* component and module. Reciprocal task interdependencies—not simply linearly pooled or sequential—result in strongly connected complex systems (Thompson, 1967).

The majority of systems research thus far has examined systems that possess a subset of these three characteristics, such as unweighted-and-directed systems (e.g., Gokpinar *et al.*, 2010) and weighted-and-directed systems (e.g., Baldwin, MacCormack, and Rusnak, 2013; MacCormack, Baldwin, and Rusnak, 2012). In addition, much of the current management work on modularity has focused on weakly connected (i.e., linear and sequential) systems such as the value chain (e.g., outsourcing areas of production) or technical product design (e.g., software) (Campagnolo and Camuffo, 2010). Only a few advanced systems papers have examined strongly connected systems (e.g., Smith and Eppinger, 1997), but have simplified the analysis to fewer than three levels due to significant analysis challenges (Smith and Eppinger, 1997).

Another challenge associated with analyzing a system's architecture is grouping components (e.g., tasks) into subsystems such that components in the *same* subsystem interact more strongly than components in *different* subsystems. A system is "composed of interrelated subsystems, each of latter being, in turn, hierarchic in structure" (Simon, 1962: 468). Grouping components into modules, also known as "classifying" (e.g., Everitt, Landau, and Leese, 2001), "graph partitioning" (e.g., Goemans and Williamson, 1995; Kannan, Vempala, and Vetta, 2004), "block-diagonalizing" (e.g., Steward, 1981), "graph coloring" (e.g., Bollobas, 2000), and finding "communities" (e.g., Newman, 2004, 2006), is nontrivial.

To this end, the method advanced in this article will allow researchers to simultaneously examine component *and* module structures of architectures that are *directed*, *weighted*, and *strongly connected*—critical features of highly complex systems. Originating in mathematics, engineering, and computer sciences, spectral graph partitioning methods permit the *grouping* of a highly complex system's components into appropriate modules or groups. Spectral graph partitioning approaches (e.g., Chung, 1997; von Luxburg, 2007), in general, and the spectral approach (e.g., Zhou, Huang, and Scholkopf, 2005) discussed in this study are applicable to many complex systems within organizations. Specifically, this method will allow for further examination of tasks that may be asymmetric and reciprocal, and face dynamic shocks. Examples of such subsystems include the recursive interdependence of R&D (research and development) with QA (quality assurance) testing in product design and the iterative process of decision making by executives within an organization who are coordinating strategies. We present this methodology using a U.S. airline's production system.

Additional potential contributions are as follows. Baldwin and Clark (1997, 2000, 2006) define three distinct and potentially significantly different architectures (i.e., design, production, and in-use) for a single artifact. First, the module boundaries of these three architectures are often different (Fourcade and Midler, 2004; MacDuffie, 2013). Second, module boundaries may be unknown at a specific point in time or "hidden" (Baldwin *et al.*, 2013; MacDuffie, 2013) due to the bounded rationality or different objectives of agents (Simon, 1962). Moreover, architectures may be in flux (i.e., era of ferment) preceding a dominant design (Henderson and Clark, 1990). Third, even when module boundaries are "visible," the system architecture may drift significantly from the "blueprint." In fact, the architecture may *emerge* over time because frequent exogenous shocks may divert the system from an "originally-designed" state to some other state. For example, in the airline industry, inclement weather and mechanical problems cause ripple effects that alter a production system's initial design (i.e., schedule). Therefore, an approach that enables management research to *uncover* system architecture (i.e., system, module, and component) structures will help to advance our understanding of complex systems.

METHODOLOGICAL INNOVATION

Although researchers have examined several systems of firms (i.e., product or technical, production, and organizational), we analyze the firm-level *production* task system of firms in the passenger airline sector. The system denotes nonstop routes from an Origin to a Destination (OD). A carrier (i.e., airline) moves passengers between airports. We use the Airline On-Time Performance (AOTP) database to illustrate the methodology described in this article. Table 1A presents sample AOTP information for one aircraft of one carrier during one week. Fourteen distinct *ODs* or *task components* are present (c.f. Table 1B). This one aircraft departed OAK (Oakland) for JFK (New York) on January 1, 2003, at 07:05. After deplaning passengers, servicing (e.g., fueling), and enplaning passengers, the *same* aircraft departed for LGB (Long Beach) on January 1, 2003, at 16:43. Note that task execution requires resource inputs—namely, an assigned aircraft—and an aircraft resource cannot be used in two tasks simultaneously. Table 1A identifies, for example, that this one aircraft creates *one* dependency between the OAK → JFK task and the JFK → LGB task because the destination of the prior flight equals the origin of the next. Thus, aircraft create sequential and reciprocal task dependencies (Barnhart *et al.*, 1998).

Traditionally, grouping of components into modules has been done by consulting external agents (e.g., designers), using external artifacts (e.g., code base directory structure), or both (e.g., Gokpinar *et al.*, 2010; MacCormack, Rusnak, and Baldwin, 2006; Sosa *et al.*, 2013). We use the alternative approach of identifying modules through detailed patterns of observed *interactions* (e.g., Baldwin *et al.*, 2013; Steward, 1981); this spectral partitioning method is presented in Table 2. Interested readers may review Chung (1997, 2005), Zhou *et al.* (2005), and von Luxburg (2007) for proofs.

System representation: vertices and their interactions

Figure 1 depicts a carrier's "runtime" (MacCormack *et al.*, 2006, 2012; MacDuffie, 2013) "flat" (Sosa *et al.*, 2013), "connection" (Barnhart *et al.*, 1998), "unsorted" (Steward, 1981) production system's strongly connected, weighted (e.g.,

DOI: 10.1002/smj

Table 1. Sample AOTP information for a carrier and one of its aircraft

(A)					(B)	
Origin	Destination	Flight date	Departure time (local)	Arrival time (local)	Task (OD)	Times performed
OAK	JFK	1/1/2003	0705	1541	FLL-IAD	1
JFK	LGB	1/1/2003	1643	1930	FLL-JFK	4
LGB	OAK	1/1/2003	2036	2157	IAD-FLL	1
OAK	IAD	1/1/2003	2315	0721	IAD-OAK	1
IAD	OAK	1/2/2003	0816	1052	JFK-FLL	4
OAK	JFK	1/2/2003	1205	2033	JFK-LGB	2
JFK	FLL	1/3/2003	1424	1742	JFK-MCO	1
FLL	JFK	1/3/2003	1825	2055	JFK-RSW	1
JFK	RSW	1/4/2003	1050	1358	LGB-JFK	1
RSW	JFK	1/4/2003	1450	1712	LGB-OAK	1
JFK	LGB	1/4/2003	1934	2159	MCO-JFK	1
LGB	JFK	1/5/2003	0700	1509	OAK-IAD	1
JFK	MCO	1/5/2003	1613	1853	OAK-JFK	2
MCO	JFK	1/5/2003	1950	2205	RSW-JFK	1
JFK	FLL	1/5/2003	2302	0213		
FLL	IAD	1/6/2003	0723	0938		
IAD	FLL	1/6/2003	1043	1306		
FLL	JFK	1/6/2003	1352	1616		
JFK	FLL	1/6/2003	1728	2026		
FLL	JFK	1/6/2003	2106	2344		
JFK	FLL	1/7/2003	0845	1133		
FLL	JFK	1/7/2003	1214	1456		

All U.S. carriers, with passenger revenue of at least one percent, file AOTP data with the U.S. Department of Transportation. Data include, for every departure, items such as OD airports, departure date, departure and arrival times, and aircraft tail number. These data represent *actual* and not necessarily scheduled events. Table B illustrates that even in this simple example for a fixed time interval (e.g., week) a carrier can perform or execute a task multiple times.

Holtta-Otto *et al.*, 2012) digraph[1] for one quarter in one year. It contains 48 components and 309 interactions. Each vertex is a task, and the last row depicts total task executions (e.g., FLL → IAD was executed 176 times). Columns are arc "tails" depicting the preceding task and rows are arc "heads" depicting the subsequent task. Each arc is a valued dependency between two components (i.e., the output of one component is an input for another). The dependency, for example, between FLL → IAD and IAD → FLL tasks is 89. This dependency results because 89 (of the possible 176) aircraft assigned to FLL → IAD are assigned to IAD → FLK. Figure 1 illustrates that the dependence of component u on v is not the same as the dependence of component v on u. Also, dependencies *vary* (e.g., 1–969). Finally, less obvious but critical is that the system is strongly connected — a path exists between every vertex pair[2] — because regulations require every aircraft be periodically routed to the carrier's few costly maintenance, repair, and overhaul (MRO) stations (Barnhart *et al.*, 1998, 2002). Figure 1 remains strongly connected even after upwards of 80 percent of the dependencies are deleted.

Graph spectra partitioning

Strongly connected systems are common in practice, but uncommon in the extant systems literature (Smith and Eppinger, 1997). One extensively-used method (e.g., Baldwin and Clark, 2000; Gebala and Eppinger, 1991) is to block-diagonalize the digraph (e.g., Steward, 1981), but *block-diagonalizing a digraph is not possible for strongly connected digraphs* (Bollobas, 2000). An alternative is to "tear" the strongly connected digraph, but tears are neither unique nor deterministic (Steward,

[1] A weighted digraph $G = (V, E)$ is: (1) a set V of vertices; (2) a subset of edges $E \subseteq V \times V$ between ordered pairs $[u,v]$ (i.e., $u \rightarrow v$) of vertices u and v; and (3) a positive real number function w: $E \rightarrow \mathbb{R}^+$ for each edge $[u,v]$ ϵ E.

[2] A path is a tuple $(v_1, \ldots, v_p)$ starting with u and ending with v or $[v_l, v_{l+1}]$ ϵ E and $1 \leq l \leq p-1$, and $v_1 = u$ and $v_p = v$.

DOI: 10.1002/smj

Table 2. Spectral graph partitioning method: overview

1. **System representation**: Create a digraph representation of carrier *i*'s time *t* operational system where *V* denotes the set of vertices and $|V|$ denotes the number of vertices[a]
2. **Graph spectra computation:**
   - 2.1. Define a random walk on the digraph (i.e., discrete time finite state Markov chain)[b]
   - 2.2. For the random walk digraph:
     - 2.2.1. Compute the unique stationary distribution of the random walk digraph[c]
     - 2.2.2. Identify the unique stationary distribution. The eigenvalue of the stationary distribution λ_s is equal to 1.0 and all values in the associated eigenvector will be positive real numbers > 0[d]
     - 2.2.3. Form the square ($|V|$ by $|V|$) diagonal matrix Π. The diagonal equals the eigenvector of the stationary distribution[e]
   - 2.3. Compute $\Theta = \frac{1}{2}(\Pi^{1/2}P^T\Pi^{-1/2} + \Pi^{-1/2}P\Pi^{1/2})$[f]
   - 2.4. Compute all eigenvalues and eigenvectors of Θ. Because Θ is symmetric, all eigenvalues and eigenvectors will be real numbers[g]
   - 2.5. Sort (descending) eigenvalues (λ_k) and label ($k = 1, \ldots, |V|$). Eigenvalues will be bounded between 1 and -1 or $\lambda_1 \approx 1$ and $\lambda_{|V|} \approx -1$
3. **Cluster generation**: For $k = 2$ to $k_{max} \leq |V|$:
   - 3.1. Create the *k* module solution using the eigenvectors associated with the *k* largest eigenvalues. For example, *k*-means[h], finite mixture models, or heuristic methods can be used to group components into modules just to name a few. *k*-means is most commonly used
   - 3.2. Repeat for all context relevant values of $2 \leq k \leq k_{max}$
4. **Cluster selection**: Select k^* (i.e., "best") using an appropriate metric. Examples are eigengap heuristic (e.g., von Luxburg, 2007), modularity metric (e.g., Newman and Girvan, 2004), AIC/BIC metric (e.g., McLachlan and Peel, 2000), and Calinski and Harabasz (1974). Metric is dependent on research, goals, and context

[a] Columns denote arcs.
[b] Step 2.1: Divide each cell by column total. Import it to Matlab. This square matrix is *P* or probability (transition) matrix.
[c] Step 2.2.1: "[lVec,eVal] = eig(P′);". eVal (i.e., eigenvalues) and lVec (*left* eigenvectors) are both square matrices.
[d] Step 2.2.2: Identify the column in eVal with entry = 1. Set column number to c. If no column exists then ***STOP***—key assumption is violated and the proposed method will render nonsensical groupings.
[e] Step 2.2.3: "Pi = diag(lVec[:,c]);".
[f] Step 2.3: "Theta = 0.50 * ((Pi^0.50 * P′ * Pi^-0.50) + (Pi^-0.50 * P * Pi^0.50));".
[g] Setp 2.4: "[thetaVec,thetaVal] = eig(Theta);". Range of the diagonal of thetaVal is [-1,1]. The square of each value of the thetaVec column (for thetaVal = 1) is the stationary distribution (i.e., "diag(Pi)") per Lemma 3.2 (Zhou *et al.*, 2005).
[h] Step 3.1: If available (Statistical Toolbox needed) and given thetaVec columns are sorted (descending by thetaVal) into matrix BASIS, execute "IDX = kmeans(BASIS(:,1:k),k)" where k (e.g., 7) is number of groups requested. IDX column vector stores vertex group membership number (i.e., [1, 7] for $k = 7$). Repeat for values of *k* as needed.

1981), and quickly become intractable (Johnson, 1975).

Spectral methods (e.g., Fiedler, 1973; Kannan *et al.*, 2004; Shi and Malik, 2000), on the other hand, allow us to overcome challenges associated with analyzing weighted, directed, and strongly connected complex systems, and thus, complement existing approaches[3] (von Luxburg, 2007; von Luxburg, Belkin, and Bousquet, 2008). Spectral methods—a family of methods that can differ by the "cut function" (e.g., ratio versus normalized cut) used, for example—are optimization problems that group components by minimizing (maximizing) interdependence *between* (*within*) modules (i.e., normalized cut). Although spectral methods provide an "approximate" solution (Chung, 1997), these methods provide "good" solutions under general conditions (von Luxburg *et al.*, 2008). We build

[3] Methods that use a symmetric distance metric (e.g., Euclidean or a sum-of-squares) are undirected graphs. They are "mathematically attractive due to their simplicity [but] are easy to fool" (Kannan *et al.*, 2004: 499). See Everitt *et al.* (2001) and Schaeffer (2007) for a review of each method's applicability, strengths, and limitations. "Wagner trees" or cladogram methods (e.g., AlGeddawy and ElMaraghy, 2013) assume unweighted graphs. Another key issue is that a strongly connected digraph may not be a tree because paths may not be unique. Methods (e.g., Baldwin *et al.*, 2013; MacCormack *et al.*, 2012) that rely on path lengths require a finite length; if a digraph is strongly connected, the "visibility" matrix (MacCormack *et al.*, 2012) is unbounded, thus the matrix does not converge. In addition, cycle detection methods (e.g., Tarjan, 1972) "return" one cycle if the digraph is strongly connected.

DOI: 10.1002/smj

	BTV-JFK	BUF-JFK	DEN-JFK	FLL-IAD	FLL-JFK	IAD-FLL	IAD-LGB	IAD-OAK	JFK-BTV	JFK-BUF	JFK-DEN	JFK-FLL	JFK-LAS	JFK-LGB	JFK-MCO	JFK-MSY
BTV-JFK									177							
BUF-JFK										419						
DEN-JFK											80					
FLL-IAD						1						172				
FLL-JFK						174						969				
IAD-FLL				89												
IAD-LGB				86												
IAD-OAK				1												
JFK-BTV					4											
JFK-BUF	1		1		111											
JFK-DEN		1			5											
JFK-FLL	5	296	35		259											
JFK-LAS	4	10	1		56											
JFK-LGB	61	1	2		141											
JFK-MCO		28	10		55											
JFK-MSY	1	4	5		14											
JFK-OAK	1	3	3		31											
JFK-ONT	3	1			3											
JFK-PBI	88	3	3		121											
JFK-ROC	1	1			34											
JFK-RSW	2	3	20		54											
JFK-SEA	4				67											
JFK-SJU		1	5		20											
JFK-SLC	2	3			4											
JFK-SYR					87											
JFK-TPA	2	64	1		67											
LAS-JFK													194			
LAS-LGB													84			
LGB-IAD							87							82		
LGB-JFK							85							256		
LGB-LAS														5		
LGB-OAK							5							127		
LGB-SLC														3		
MCO-JFK															661	
MSY-JFK																89
OAK-IAD								2								
OAK-JFK								123								
OAK-LGB								51								
ONT-JFK																
PBI-JFK																
ROC-JFK																
RSW-JFK																
SEA-JFK																
SJU-JFK																
SLC-JFK																
SLC-LGB																
SYR-JFK																
TPA-JFK																
	175	419	86	176	1133	175	177	176	177	419	80	1141	278	473	661	89

	JFK-OAK	JFK-ONT	JFK-PBI	JFK-ROC	JFK-RSW	JFK-SEA	JFK-SJU	JFK-SLC	JFK-SYR	JFK-TPA	LAS-JFK	LAS-LGB	LGB-IAD	LGB-JFK	LGB-LAS	LGB-OAK
BTV-JFK																
BUF-JFK																
DEN-JFK																
FLL-IAD																
FLL-JFK																
IAD-FLL													87			
IAD-LGB													88			
IAD-OAK																
JFK-BTV											1			3		
JFK-BUF											8			6		
JFK-DEN											2			3		
JFK-FLL											26			76		
JFK-LAS											78			19		
JFK-LGB											4			66		
JFK-MCO											63			45		
JFK-MSY											9			8		
JFK-OAK											18			14		
JFK-ONT											1			5		
JFK-PBI											16			93		
JFK-ROC											3			1		
JFK-RSW											37			20		
JFK-SEA														3		
JFK-SJU											3			87		
JFK-SLC														4		
JFK-SYR											2			7		
JFK-TPA											6			14		
LAS-JFK															82	
LAS-LGB															157	
LGB-IAD												4				
LGB-JFK												101				
LGB-LAS												15				
LGB-OAK												117				
LGB-SLC												1				
MCO-JFK																
MSY-JFK																
OAK-IAD	88															85
OAK-JFK	221															2
OAK-LGB	37															514
ONT-JFK		79														
PBI-JFK			630													
ROC-JFK				338												
RSW-JFK					364											
SEA-JFK						71										
SJU-JFK							260									
SLC-JFK								50								
SLC-LGB																
SYR-JFK									178							
TPA-JFK										406						
	346	79	630	338	364	71	260	50	178	406	277	238	175	474	239	601

	LGB-SLC	MCO-JFK	MSY-JFK	OAK-IAD	OAK-JFK	OAK-LGB	ONT-JFK	PBI-JFK	ROC-JFK	RSW-JFK	SEA-JFK	SJU-JFK	SLC-JFK	SLC-LGB	SYR-JFK	TPA-JFK
BTV-JFK																
BUF-JFK																
DEN-JFK																
FLL-IAD																
FLL-JFK																
IAD-FLL																
IAD-LGB				1												
IAD-OAK				172												
JFK-BTV		85			1		2			80	1					
JFK-BUF		50			12		2	101	5	10	1	15	1			94
JFK-DEN		1			62			3	6	4						
JFK-FLL		113			71		4	103	8	58	1	32	11		2	26
JFK-LAS		19			11		5	28	3	8	6	18	2		1	9
JFK-LGB		18	2		93			40	1	15		9	4		1	16
JFK-MCO		59	79		24		7	31	88	9	4	47	4		86	18
JFK-MSY		4			4			14	4	2		10	1			6
JFK-OAK		6	1		17		44	9	60	7	10	75	24			12
JFK-ONT		2			1			1		65					1	3
JFK-PBI		36	5		17		3	89	71	41	3	20	1		1	17
JFK-ROC		13			1		1	104		3		4				172
JFK-RSW		87	1		21		10	62	1	10	4	10	6		1	8
JFK-SEA		1			1			4		5						2
JFK-SJU		94			5		1	18	2	1		20				8
JFK-SLC		67						1	1	2					2	3
JFK-SYR		4			2		6	4		1	55	2	3			3
JFK-TPA		8	1		6		2	15	88	40	1	3	1		82	6
LAS-JFK																
LAS-LGB																
LGB-IAD						2										
LGB-JFK						23								11		
LGB-LAS						140								77		
LGB-OAK						353										
LGB-SLC						85										
MCO-JFK																
MSY-JFK																
OAK-IAD																
OAK-JFK																
OAK-LGB																
ONT-JFK																
PBI-JFK																
ROC-JFK																
RSW-JFK																
SEA-JFK																
SJU-JFK																
SLC-JFK																
SLC-LGB	85															
SYR-JFK																
TPA-JFK																
	85	667	89	173	349	603	87	627	338	361	86	265	58	88	177	403

Figure 1. "Runtime" "flat" production system example. The last row denotes column totals. Dividing each cell by the column total transforms the digraph to a discrete time finite state Markov chain.

DOI: 10.1002/smj

on Smith and Eppinger (1997)—namely, eigendecomposition of matrices.[4]

Random walk and Markov chains

Dividing each cell value by its column sum in the Figure 1 digraph transforms it into a "random walk" or a discrete finite state *Markov chain* (c.f. Bollobas, 2000). Each cell denotes a pairwise component interaction probability (Clarkson, Simons, and Eckert, 2004).[5] A Markov chain describes a system's dynamic behavior. The dependency between FLL → IAD and IAD → FLL tasks is 89. Dividing 89 by 176 is 0.51, or a 51 percent dependence probability between the two.

Stationary distribution

A discrete finite state Markov chain may have a unique stationary probability distribution where every value is a real positive number (Bollobas, 2000; Chung, 1997). The stationary distribution is computed from the eigendecomposition of the transition matrix—the eigenvector associated with $\lambda = 1$. A large value in the stationary distribution vector suggests that a component u is more critical to a system (Smith and Eppinger, 1997).

Spectral method and its extension to digraphs

A traditional *undirected* spectral specification minimizes a "*normalized cut*"[6] objective function (Chung, 1997; von Luxburg, 2007; Shi and Malik, 2000) using eigendecomposition. The eigenvalues are sorted and the eigenvectors associated with the first k eigenvalues are used to form k groups (Fiedler, 1973; Shi and Malik, 2000). Spectral methods are not new, but a recent extension overcomes the challenges associated with imaginary (i.e., $\sqrt{-1} = i$) values. This advancement extends the method to digraphs. The eigendecomposition of the theta functional (i.e., Θ)[7] provides $|V|$ *real* eigenvalues and eigenvectors (Chi *et al.*, 2009; Zhou *et al.*, 2005). Eigenvectors are the bases, and each vertex is a "point" in the subspace spanned by the vectors. After sorting (descending) the eigenvalues, a k group solution is generated from the k eigenvalue-vectors pairs (Chi *et al.*, 2009; Zhou *et al.*, 2005).

Cluster generation and selection: finding k^*

Many methods such as finite mixture (e.g., McLachlan and Peel, 2000), heuristic (e.g., Zhou *et al.*, 2005), and principle component (e.g., Jolliffe, 2002) can be used to assign vertices to k groups. The "k-means" method is most often used (von Luxburg, 2007).[8]

Figure 2 depicts the six module solution of the Figure 1 system. Each module represents one of six possible aircraft routings; in other words, an aircraft will be used (for M_1) in *LGB → SLC*, JFK → LAS, *LGB → LAS*, *SLC → LGB*, *LGB → OAK*, *LAS → LGB*, *OAK → LGB*, and LAS → JFK. Note that with exception of JFK → LAS and LAS → JFK, all the remaining ODs are western ODs. Aircraft remain geographically proximate. These ODs exhibit mostly *within* module dependencies; thus, M_1 is a subset of this carrier's western operations. The remaining ODs are the module interfaces. JFK → LAS (i.e., east → west) is a M_1 "input" and LAS → JFK (i.e., west → east) is an "output" to other modules.

We can abstract Figure 2 to the module level (c.f. Table 3 and first column of Table 4). First, module sizes [0.04, 0.26] and between module coupling [0.00, 0.03] vary, and within module coupling is high (0.70 and 0.08 average and standard deviation, respectively). M_2, M_4, and M_6 are this firm's "core" system and the remainders are peripheral in both size and interdependence. M_3 (i.e., JFK-SEA … SEA-JFK) is the least coupled. JFK-SEA and SEA-JFK are interesting. At the time, this carrier's U.S. Northwest passenger traffic was small (i.e., π(JFK-SEA) = π(SEA-JFK) = 0.58). In contrast, this carrier emphasized the eastern United States—specifically, New York and Florida. M_2,

---

4 $\mathbf{AX} = \mathbf{X\Lambda}$. Columns of $\mathbf{X}$ and $\mathbf{\Lambda}$ are the eigenvectors and values, respectively.

5 Each component is one of the possible |V| "states" and each cell is the transition probability p: $V \times V \rightarrow \mathbb{R}^+$. Each cell is $p([u,v]) = w([u,v])/d(u)$ for all $[u,v] \in E$ and 0 otherwise where $d(u) = \sum_{u \rightarrow v} w([u,v])$.

6 A cut is those edges (and associated weights) that span groups. The objective function is $\min \sum (vol\partial S_k / vol S_k)$. The size or volume of module S is vol$S = \sum_{u \in S} \pi(u)$. The boundary of module S is $\partial S = \{[u,v] \mid u \in S \text{ and } v \notin S\}$. Module S's coupling with all other modules is vol$\partial S = \sum_{[u,v]} \pi(u)\, p(u,v)$ for every $[u,v] \in \partial S$.

7 Per Chung (2005) and Zhou *et al.*, (2005), $\Theta = \frac{1}{2}(\Pi^{1/2} P \Pi^{-1/2} + \Pi^{-1/2} P^T \Pi^{1/2})$, where Π and P are the diagonal stationary and transition matrices, respectively.

8 Eigenvectors are orthogonal, thus k-means limitations are minimized (e.g., Zha *et al.*, 2001).

DOI: 10.1002/smj

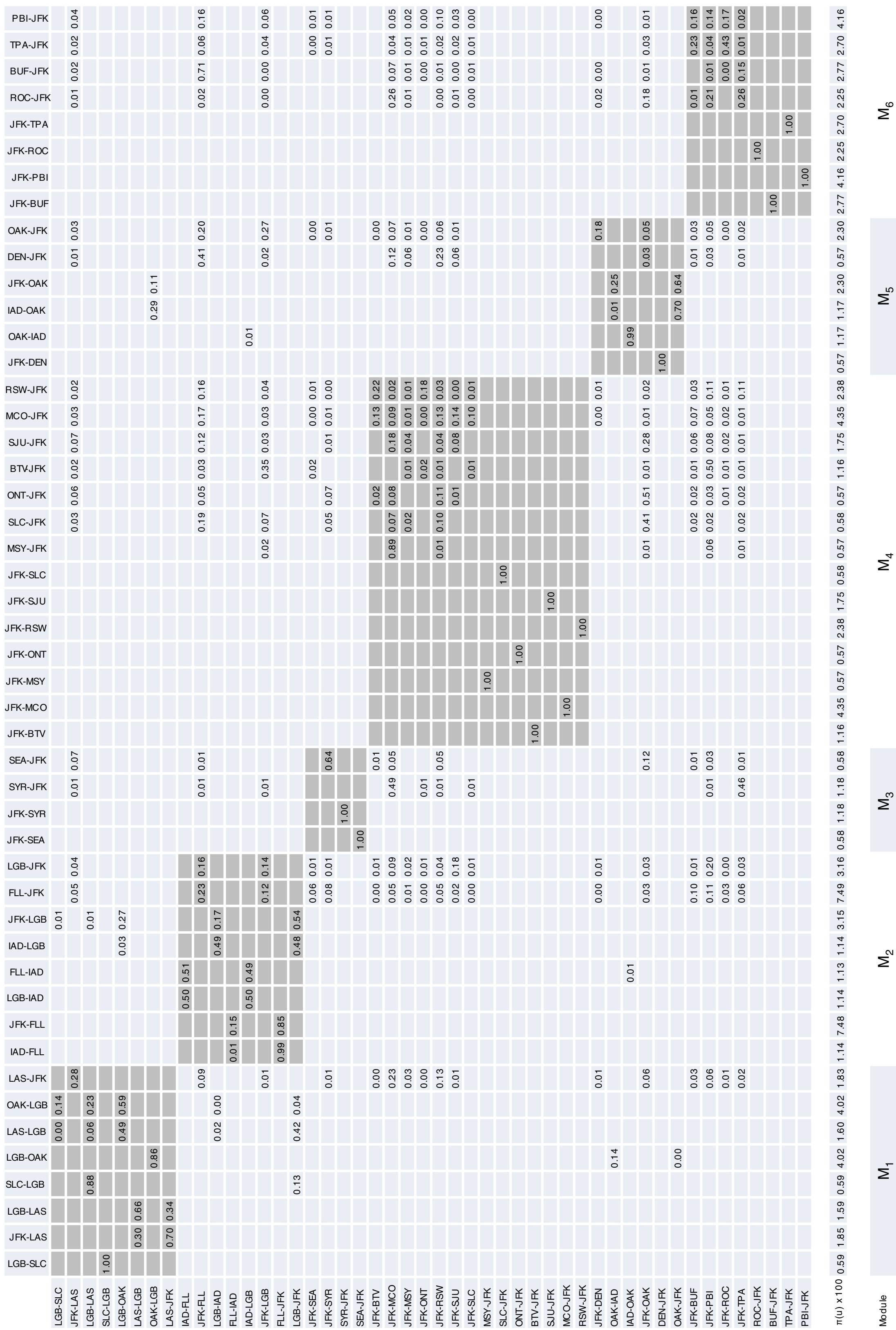

Figure 2. Production system subsystems (modules). Six modules denoted in the figure. The last row denotes the stationary distribution for each OD.

Table 3. Production subsystem descriptives as depicted in Figure 2

Module	Size[a] ($volS_j$)	Total coupling[b] ($vol\partial S_j$)	Cut[c] ($vol\partial S_j$ /$volS_j$)	Between module coupling[d] (interdependence) M_1	M_2	M_3	M_4	M_5	M_6		
M_1	0.16	0.03	0.18		0.01	0.00	0.00	0.01	0.00		0.03
M_2	0.26	0.08	0.31	0.01		0.00	0.02	0.01	0.03		0.08
M_3	0.04	0.01	0.40	0.00	0.01		0.00	0.00	0.00		0.01
M_4	0.23	0.06	0.28	0.01	0.02	0.01		0.01	0.02		0.06
M_5	0.08	0.03	0.36	0.01	0.00	0.00	0.01		0.01		0.03
M_6	0.24	0.06	0.27	0.00	0.03	0.01	0.02	0.00			0.06
											↑
	1.00			0.03	0.08	0.01	0.06	0.03	0.06	←	Total

Total cut = 1.80.
[a] $volS_j$: Size or sum of the module's components' stationary probabilities (last row). Size(M_3) = (0.58 + 1.18 + 1.18 + 0.58) ÷ 100 = 0.04. Factor of 100 is present because stationary distributions in Figure 4 are ×100.
[b] $vol\partial S_j$: Total coupling (interface) with *other* modules—namely, the extent to which other modules depend on this modules for input. First sum probabilities of arcs outside of module or JKF → SEA(0.00), JFK → SYR(0.00), SYR → JFK(1.00), and SEA → JFK(0.36). Sum of the product of each sum multiplied by the stationary probability. Total coupling(M_3) = [(0.00·0.58) + (0.00·1.18) + (1.00·1.18) + (0.36·0.58)] ÷ 100 = 0.01.
[c] Cut: Ratio of total coupling with other modules ÷ module size ($vol\partial S_j \div volS_j$) or how coupled is this module with other modules, controlling for the module size. Cut(M_3) = 0.01 ÷ 0.04 ≈ 0.40.
[d] Coupling(M_a, M_b): Between module coupling (pair-wise interdependence). Coupling (M_2, M_6) = 0.03 or [(0.10 + 0.11 + 0.03 + 0.06)·7.49] ÷ 100 + [(0.01 + 0.20 + 0.00 + 0.03)·3.16] ÷ 100 = 0.01 + 0.02—namely, the amount of coupling where components of M_6 depend on the outputs from components of M_2.

Table 4. Carrier's system architectural descriptives

Time	2003 Q1	2008 Q1	2014Q1
Vertices (tasks)	48	224	282
Dependencies	309	1,777	2,723
System modularity (Q)	0.52	0.58	0.62
Modules (k^*)	6	8	10
Average tasks per module	8.00	28.00	28.20
Module size ($volS_j$) range	[0.04,0.26]	[0.02,0.24]	[0.02,0.21]
Module size S.D. ($\sigma(volS_j)$) × 100	9.15	6.50	5.84
Average within module coupling (1.0 – ($vol\partial S_j/volS_j$))	0.70	0.73	0.75
Within module coupling S.D.	0.08	0.09	0.07
Between module coupling range ($vol\partial S_j$)	[0.00,0.03]	[0.00,0.02]	[0.00,0.01]
Between module coupling S.D. ($\sigma(vol\partial S_j)$) × 100	0.91	0.47	0.34

M_4, and M_6 are significant proportion of this carrier's traffic. M_2 contains FLL (Fort Lauderdale, FL), M_4 contains RSW (Fort Myers, FL) and MCO (Orlando, FL), and M_6 contains PBI (Palm Beach, FL). This carrier divided its key Florida operations into manageable "chunks."

Finally, the "k-means" method identifies an architecture that contains k modules. Determining "best" k^* is an extensive body of research.[9] In this study, we use the Q modularity statistic (Newman, 2004, 2006; Newman and Girvan, 2004). Q is 1 if the system is perfectly modular and $Q < 1$ for less modular systems. Although many modularity metrics have been used (see Holtta-Otto *et al.*, 2012 for a review), White and Smyth (2005) provide an analytic solution regarding the "behavior" of the Q metric. Figure 3 plots Q for each value of k for the system depicted in Figure 1. Q is a maximum (i.e., $Q = 0.52$) when $k^* = 6$.

[9] For $k > k'$, k is a more granular decomposition (AlGeddawy and ElMaraghy, 2013; Chiriac *et al.*, 2011). Many methods such as statistical (e.g., Calinski and Harabasz, 1974), information theoretic (e.g., Sugar and James, 2003), and heuristic (e.g., Kernighan and Lin, 1970) approaches exist. Finite mixture models provide a metric (i.e., AIC) that permit the comparison between values of k. Identifying the appropriate level of decomposition is associated with *all* graph partitioning algorithms—not just spectral (von Luxburg, 2007; Sugar and James, 2003).

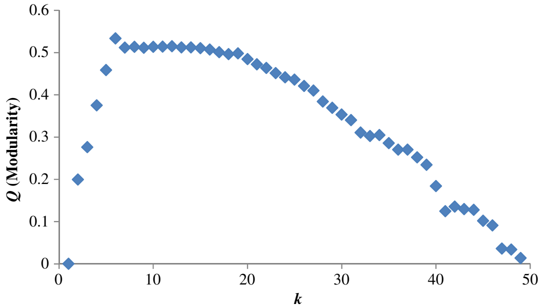

Figure 3. System modularity (Q) for different values of k. Let $Q(P_k) = \sum_{1 \le j \le k} \left[A(S_j, S_j) - A(S_j, S)^2 \right]$, where $P_k = S_1 \ldots \cup S_k$, $S_j \cap S_{j'} = \emptyset$ for $j \neq j'$ and $0 \le Q(P_k) \le 1$. Q decreases as the system becomes *less* modular. The term $A(S_j, S_j)$ denotes the within module sum of weights or ($\mathrm{vol}S_j - \mathrm{vol}\partial S_j$) and the term $A(S_j, V)$ denotes the sum of weights over all edges attached not vertices in S_j or $\mathrm{vol}S_j$. The Newman and Girvan (2004) specification includes, in the denominator of both terms in the summation, total digraph weights or $A(V, V)$. We omit this term because it is 1 in the study's context. For $k^* = 6$ depicted in Figure 2, $Q = 0.52$. "Realistic" system Q is $0.3 \le Q \le 0.7$ (Newman and Girvan, 2004; White and Smyth, 2005). An alternative modularity metric is the total normalized cut (i.e., $\sum vol\partial S_j / volS_j$). *Small* values are associated with modular architectures.

Table 4 presents architectural metrics for this carrier in 2003Q1, 2008Q1, and 2014Q1. Despite an almost *sextupling* and *order-of-magnitude* increase in tasks and dependencies, respectively, modularity *increases*. The number of modules increases (6 to 10), but other subtleties exist. First, the number of tasks per module increases (8.00 to 28.20), but maximum module size decreases (0.26 to 0.21). Second, modules are "rebalanced" to be more similar in size (9.15 to 5.84). Third, modules are more cohesive (within module coupling is 0.70 to 0.75) and less variable (0.08 to 0.07). Finally, between module coupling (0.03 to 0.01) and their variability (0.91 to 0.34) decrease. In short, this brief examination of a few simple metrics helps to illustrate one of many potential approaches for managing complexity not only at a given time but also across time. This carrier's modularization process (MacDuffie, 2013) and the efficacy (e.g., economic performance) of this carrier's approach, of course, require additional empirical analysis.

DISCUSSION AND CONCLUSION

In this study, we introduce to management research a spectral partitioning methodology and one specific extension that permit the architectural determination of a complex system containing *strongly connected*, *weighted*, and *directed* interactions. We illustrate the method using the U.S. passenger airline sector. Although such complex systems are common (Smith and Eppinger, 1997), studies examining them are still rare. Applying spectral methods should enable researchers to advance our understanding of the behavior, management, and outcomes associated with complex systems, which is important, given the ubiquity of complex systems—be they industries, groups of firms, firms, systems within firms, or products. Indeed, the proposed method is tractable and implemented using a few commands (c.f. Table 2 for sample Matlab© code).

To further compare the spectral method to an existing method, we analyzed the system in Figure 1 with a heuristic path-based method (e.g., Steward, 1981). Path-based methods *require* a graph that is *not* strongly connected. After repeatedly "tearing" (by deleting arcs) via the shortest path identified by a transitive closure (e.g., Dijkstra) method, we formed modules using k^* (maximum Q). The architecture, presented in Figure 4, contains 21 modules. Deleting 16 arcs, found in the "lower triangle," block-diagonalizes the digraph. Twenty modules contain *one* task (e.g., JFK → PBI). The one-task modules are expected because for these tasks (e.g., JFK → PBI), tearing the single arc (e.g., between JFK → PBI and PBI → JFK) leaves the former with *no* successor. By design, repeated tears repeatedly separate a few tasks from the system. Decoupling one task from the system leaves the remainder intact. The *last* module contains the remaining 28 strongly connected tasks (i.e., JFK → BTV to LGB → JFK). In short, tearing identifies too many modules that are inconsistent with the problem context. This approach is also more computationally demanding than the spectral approach presented herein.

Despite the potential of spectral methods, we reiterate that grouping vertices on general graphs is NP-*hard* (c.f. Fortnow, 2009; Karp, 1972). In general, spectral methods are "approximations"—they "relax" the NP-*hard* problem (Chung, 1997; von Luxburg, 2007). To the best of our knowledge, no alternatives exist for strongly connected digraphs—except "brute-force" enumeration, which also becomes quickly intractable because "the number of elementary circuits in a directed graph can grow *faster* … than the exponential 2^n"

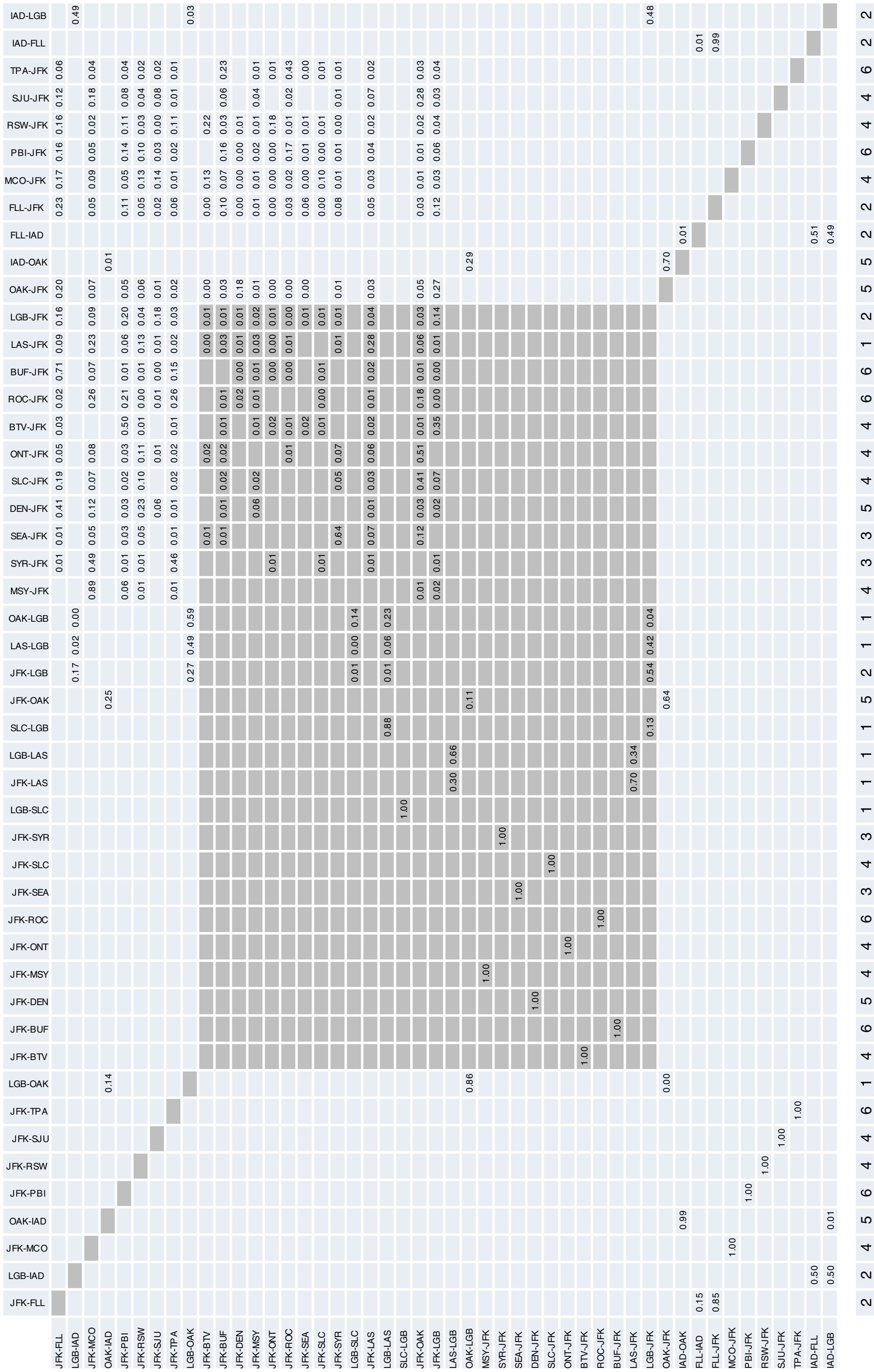

Figure 4. Production system modularization using heuristic path-based method. The last row denotes each OD's module number (M_1 – M_6) found in Figure 2.

DOI: 10.1002/smj

(Johnson, 1975: 77, emphasis added). Spectral methods, however, behave predictably (c.f. Kannan *et al.*, 2004; von Luxburg *et al.*, 2008). If spectral analysis is a precursor to empirical estimation, this predictable behavior or "error" (e.g., number of modules) can be included as estimation controls.

Furthermore, we caution again that the presented method *requires* a unique stationary distribution. If one does not exist, the proposed methodology is inappropriate. In short, the proposed method *complements* other methods (e.g., Baldwin *et al.*, 2013).[10] A one-size-fits-all approach does not exist because one innocuous assumption (e.g., strongly connected) can render an approach inappropriate. We suggest that future studies should report graph properties—akin to sample descriptive statistics associated with regression analyses (c.f. Table 4). Nonetheless, even if the entire digraph is not strongly connected, it is likely that a strongly connected *sub*system exists (Smith and Eppinger, 1997), which could be analyzed using spectral methods.

The strongly connected assumption, however, suggests a system property that may have been overlooked. The largest eigenvalue can be $\lambda < 1$, $\lambda = 1$ (stationary), or $\lambda > 1$. The system "decays" if $\lambda < 1$ and grows without bounds or "explodes" if $\lambda > 1$. What is desirable in one context may be undesirable in another. For example, a project's lambda should be less than 1; otherwise, the project does not end (i.e., is "unstable"). In contrast, a software user base's lambda should be greater than 1 because in-use architecture "instability" creates network effects. Even within a given architecture (e.g., product versus transportation network design), variation may exist. Hence, spectral methods may be used to categorize and compare different system "classes."

The spectral method presented in this study is especially useful if the architecture is "hidden" (Baldwin *et al.*, 2013). Analyzing seventeen software projects[11], Baldwin *et al.* (2013) determined the visible design architecture using the code's directory structure because it "reflects both programming conventions and the designers' intuition" (p. 7). The authors then traced "function calls" and determined the design architecture from the code. They found significant differences, thus suggesting an architecture "hidden" from designers. To this end, spectral methods can be used to quantify discrepancies between the visible and hidden architectures. The eigenvalue distribution (i.e., spectra) is a "fingerprint." One of the best-known is the "semi-circle" for a random undirected binary graph (Bollobas, 2000). A discrepancy between the visible and hidden architectures may be quantifiable by comparing the two spectra (e.g., nonparametric statistic). Two different architectures (e.g., design vs. in-use) may also be compared if one or both architectures exceed the designers' bounded rationality, are in flux, or both. Although all architectures are important (Baldwin and Clark, 2000), the runtime architecture is likely to be significantly different from the design architecture (MacCormack *et al.*, 2006, 2012). In fact, irregular airline operations (e.g., inclement weather recovery) are the focus of significant work (Belobaba, Odoni, and Barnhart, 2009). Spectral methods can offer insights into the modularization process (Brusoni and Prencipe, 2011; MacDuffie, 2013) and help assess their success. Using spectral methods, scholars could examine modularizations that results from different modularization techniques and how they change over time.

Finally, spectral methods can contribute beyond the study of architectures. Examples include studying systems of tasks between R&D and quality assurance units, shared tasks between executives in different divisions of the firm, and tasks between alliance partners in an alliance network. Furthermore, Chen, Su, and Tsai (2007) argue that "competitive relationships ['dependencies'] between a pair of firms can be asymmetric" (p. 116) and that "competition occurs at multiple levels" (p. 115). Vertices can denote firms and arcs can denote multimarket commonality (e.g., Chen, 1996). Spectral methods may also help provide insights to questions in the strategic groups research (e.g., Hatten and Hatten, 1987; McNamara, Deephouse, and Luce, 2003) of whether firms compete more with rivals from inside or outside a strategic group.

We hope that a tractable methodology will motivate follow-on studies. Arguably, conceptual advances may have out-paced empirical advances.

[10] A strongly connected weighted digraph may not contain a stationary distribution. If so, then to the best of our knowledge, the problem must be simplified (e.g., Condon and Karp, 2001), a heuristic method (e.g., Kernighan and Lin, 1970) used, or both. Heuristic or "greedy" methods, however, can behave unpredictably.

[11] Software components should not be strongly connected. The environment (e.g., Unix) passes control to a module. It opens files and initializes virtual memory and variables—namely, *no* predecessor. At least one module will close files, release memory, pass control back to the environment, and gracefully terminate—namely, *no* successor.

This article's methodological innovation may help narrow that gap. Our presented method may provide more reliable answers to a key management question—the relationship between architecture and performance—but may also address new questions pertinent to managing complexity in dynamic environments.

ACKNOWLEDGEMENTS

We gratefully acknowledge the valuable contributions of Shusen Liu, John Shortle, Dirk Weiss, and seminar participants at George Mason University Center for Air Transportation Systems Research (CATSR); the guest editors Michelle Gittelman and Nicolaj Siggelkow; and our anonymous reviewers. We especially appreciate the key assistance of Tiago Requeijo and John Sherry. The authors are listed in random order.

REFERENCES

Alexander A. 1964. *Notes on the Synthesis of Form*. Harvard University Press: Cambridge, MA.

AlGeddawy T, ElMaraghy H. 2013. Optimum granularity level of modular product design architecture. *Manufacturing Technology* **62**: 151–154.

Baldwin CY, Clark KB. 1997. Managing in an age of modularity. *Harvard Business Review* **75**(5): 84–93.

Baldwin CY, Clark KB. 2000. *Design Rules: The Power of Modularity*. MIT Press: Cambridge, MA.

Baldwin CY, Clark KB. 2006. The architecture of participation: does code architecture mitigate free riding in the open source development model? *Management Science* **52**: 1116–1127.

Baldwin CY, MacCormack A, Rusnak J. 2013. "Hidden" structure: using network methods to map system architecture. Working paper 13-093, Harvard Business School: Boston, MA.

Barnhart C, Boland NL, Clarke LW, Johnson EL, Nemhauser GL, Shenoi RG. 1998. Flight string models for aircraft fleeting and routing. *Transportation Science* **32**: 208–220.

Barnhart C, Kniker TS, Lohatepanont M. 2002. Itinerary-based airline fleet assignment. *Transportation Science* **36**: 199–217.

Belobaba P, Odoni A, Barnhart C. 2009. *The Global Airline Industry*. John Wiley: West Sussex, UK.

Bollobas B. 2000. *Modern Graph Theory*. Springer: New York.

Brusoni S, Prencipe A. 2011. Pattern of modularization: the dynamics of production architecture in complex systems. *European Management Review* **8**: 67–80.

Calinski T, Harabasz J. 1974. A dendrite method for cluster analysis. *Communications in Statistics* **3**: 1–27.

Campagnolo D, Camuffo A. 2010. The concept of modularity in management studies: a literature review. *International Journal of Management Review* **12**: 259–283.

Chen M. 1996. Competitor analysis and interfirm rivalry. *Academy of Management Review* **21**: 100–134.

Chen MJ, Su K, Tsai W. 2007. Competitive tension: the awareness-motivation-capability perspective. *Academy of Management Journal* **50**: 101–118.

Chi Y, Song X, Zhou D, Hino K, Tseng BL. 2009. On evolutionary spectral clustering. *ACM Transactions on Knowledge Discovery from Data* **3**: 1–30.

Chiriac N, Holtta-Otto K, Lysy D, Suh ES. 2011. Levels of modularity and different levels of system granularity. *Journal of Mechanical Design* **133**: 101007–101016.

Chung F. 1997. *Spectral Graph Theory*, CBMS Regional Conference Series in Mathematics (Volume **92**). American Mathematical Society: Providence, RI.

Chung F. 2005. Laplacians and the Cheeger inequality for directed graphs. *Annals of Combinatorics* **9**: 1–19.

Clarkson PJ, Simons C, Eckert C. 2004. Predicting change propagation in complex design. *Journal of Mechanical Design* **126**: 788–797.

Condon A, Karp RM. 2001. Algorithms for graph partitioning on the planted partition model. *Random Structures and Algorithms* **18**: 116–140.

Eisenhardt KM, Martin JA. 2000. Dynamic capabilities: what are they? *Strategic Management Journal* **21**: 1105–1121.

Ethiraj SK, Levinthal D. 2004. Modularity and innovation in complex systems. *Management Science* **50**: 159–173.

Everitt BS, Landau S, Leese M. 2001. *Cluster Analysis*. Arnold: London, UK.

Fiedler M. 1973. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal* **23**: 298–305.

Fixson SK, Park J. 2008. The power of integrality: linkages between product architecture, innovation, and industry structure. *Research Policy* **37**: 1296–1316.

Fortnow L. 2009. The status of the P versus NP problem. *Communications of the ACM* **52**: 78–86.

Fourcade F, Midler C. 2004. Modularization in the auto industry: can manufacturer's architectural strategies meeting supplier's sustainable profit trajectories? *International Journal of Automotive Technology and Management* **4**: 240–260.

Furlan A, Cabigiosu A, Camuffo A. 2014. When the mirror gets misted up: modularity and technological change. *Strategic Management Journal* **35**: 789–807.

Galunic DC, Eisenhardt KM. 2001. Architectural innovation and modular corporate forms. *Academy of Management Journal* **44**: 1229–1249.

Gebala DA, Eppinger SD. 1991. Methods for analyzing design procedures. *ASME Design Theory and Methodology Conference*, pp. 227–233. Miami, FL.

Goemans MX, Williamson DP. 1995. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM* **42**: 1115–1145.

Gokpinar B, Hopp WJ, Iravani SMR. 2010. The impact of misalignment of organizational structure and product architecture on quality in complex product development. *Management Science* **56**: 468–484.

Hatten KJ, Hatten ML. 1987. Strategic groups, asymmetrical mobility barriers, and contestability. *Strategic Management Journal* **8**: 329–342.

Henderson RM, Clark KB. 1990. Architectural innovation: the reconfiguration of existing product technologies and the failure of established firms. *Administrative Science Quarterly* **35**: 9–30.

Hoetker G. 2006. Do modular products lead to modular organizations? *Strategic Management Journal* **27**: 501–518.

Hoetker G, Swaminathan A, Mitchell W. 2007. Modularity and the impact of buyer-supplier relationships on the survival of suppliers. *Management Science* **53**: 178–191.

Holtta-Otto K, Chiriac NA, Lysy D, Suh ES. 2012. Comparative analysis of coupling modularity metrics. *Journal of Engineering Design* **23**: 790–806.

Johnson DB. 1975. Finding all the elementary circuits of a directed graph. *SIAM Journal of Computing* **41**: 77–84.

Jolliffe IT. 2002. *Principal Component Analysis*. Springer: New York.

Kannan R, Vempala S, Vetta A. 2004. On clusterings: good, bad and spectral. *Journal of the ACM* **51**: 497–515.

Karim S. 2006. Modularity in organizational structure: the reconfiguration of internally developed and acquired units. *Strategic Management Journal* **27**: 799–823.

Karp KM. 1972. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, Miller RE, Thatcher JW (eds). Plenum: New York; 85–103.

Kernighan BW, Lin S. 1970. An efficient heuristic procedure for partitioning graphs. *Bell Systems Technical Journal* **49**: 76–80.

Lomi A, Pattison P. 2006. Manufacturing relations: an empirical study of the organization of production across multiple networks. *Organization Science* **17**: 313–332.

von Luxburg U. 2007. A tutorial on spectral clustering. *Statistics and Computing* **17**: 395–416.

von Luxburg U, Belkin M, Bousquet O. 2008. Consistency of spectral clustering. *Annals of Statistics* **36**: 555–586.

MacCormack A, Baldwin CY, Rusnak J. 2012. Exploring the duality between product and organizational structures: a test of the 'mirroring' hypothesis. *Research Policy* **41**: 1309–1324.

MacCormack A, Rusnak J, Baldwin CY. 2006. Exploring the structure of complex software designs: an empirical study of open source and proprietary code. *Management Science* **52**: 1015–1030.

MacDuffie JP. 2013. Modularity—as property, modularization-as-process, and "modularity"—as frame: lessons from the production architecture initiatives in the global automotive industry. *Global Strategy Journal* **3**: 8–40.

McLachlan G, Peel D. 2000. *Finite Mixture Models*. Wiley: New York.

McNamara G, Deephouse DL, Luce RA. 2003. Competitive positioning within and across a strategic group structure: the performance of core, secondary, and solitary firms. *Strategic Management Journal* **24**: 161–181.

Mihm J, Loch C, Huchzermeier A. 2003. Problem-solving oscillations in complex engineering projects. *Management Science* **46**: 733–750.

Newman MEJ. 2004. Fast algorithm for detecting community structure in networks. *Physical Review E* **69**: 66133–66137.

Newman MEJ. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences* **103**: 8577–8582.

Newman MEJ, Girvan M. 2004. Finding and evaluating community structure in networks. *Physical Review E* **69**: 26113–26127.

Pil FK, Cohen SK. 2006. Modularity: implications for imitation, innovation, and sustained advantage. *Academy of Management Review* **31**: 995–1011.

Rivkin JW, Siggelkow NJ. 2007. Patterned interactions in complex systems: implications for exploration. *Management Science* **53**: 1068–1085.

Sanchez R, Mahoney JT. 1996. Modularity, flexibility, and knowledge management in product and organization design. *Strategic Management Journal* **17**: 63–76.

Schaeffer SE. 2007. Graph clustering. *Computer Science Review* **1**: 27–64.

Schilling MA, Phelps CC. 2007. Interfirm collaboration networks: the impact of large-scale network structure on firm innovation. *Management Science* **53**: 1113–1126.

Shi J, Malik J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**: 888–905.

Simon HA. 1962. The architecture of complexity. *Proceedings of the American Philosophical Society* **106**: 467–482.

Smith RP, Eppinger SD. 1997. Identifying controlling features of engineering design iteration. *Management Science* **43**: 276–293.

Sosa ME, Eppinger SD, Rowles CM. 2004. The misalignment of product architecture and organizational structure in complex product development. *Management Science* **50**: 1674–1689.

Sosa ME, Mihm J, Browning TR. 2013. Linking cyclicality and product quality. *Manufacturing and Service Operations Management* **15**: 473–491.

Staudenmayer N, Tripsas M, Tucci CL. 2005. Interfirm modularity and its implications for product development. *Journal of Product Innovation Management* **22**: 303–321.

Steward DV. 1981. *Systems Analysis and Management*. Petrocelli: New York.

Sugar CA, James GM. 2003. Finding the number of clusters in a dataset: an information-theoretic approach. *Journal of the American Statistical Society* **98**: 750–763.

Tarjan R. 1972. Depth-first search and linear graph algorithms. *SIAM Journal of Computing* **1**: 146–160.

Thompson JD. 1967. *Organizations in Action*. McGraw-Hill: New York.

Ulrich K. 1995. The role of product architecture in the manufacturing firm. *Research Policy* **24**: 419–440.

Ulrich K, Ellison DJ. 1999. Holistic customer requirements and the design-select decision. *Management Science* **45**: 641–658.

White S, Smyth P. 2005. A spectral clustering approach to finding communities in graph. *SIAM International Conference on Data Mining*, pp. 274–285. Newport Beach, CA.

Worren N, Moore K, Cardona P. 2002. Modularity, strategic flexibility, and firm performance: a study of the home appliance industry. *Strategic Management Journal* **23**: 1123–1140.

Zha H, He X, Ding C, Gu M, Simon HD. 2001. Spectral relaxation for K-means clustering. *Neural Information Processing Systems* **14**: 1057–1064.

Zhou D, Huang J, Scholkopf B. 2005. Learning from labeled and unlabeled data on a directed graph. *Proceedings of the 22nd International Conference on Machine Learning*. Bonn, Germany.