Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
IDETC/CIE 2018
August 26-29, 2018, Quebec City, Quebec, Canada

DETC2018-85759

DATA-DRIVEN FUNCTION NETWORK ANALYSIS FOR PRODUCT PLATFORM PLANNING: A CASE STUDY OF SPHERICAL ROLLING ROBOTS

Binyang Song[1], Jianxi Luo[1,2], Rajesh Elara Mohan[1], Kristin L. Wood[1,2]
[1]Engineering Product Development Pillar, Singapore University of Technology and Design
[2]SUTD-MIT International Design Centre Singapore

ABSTRACT
A properly designed product-system platform can reduce the cost and lead-time to design and develop a product family and thus achieve the tradeoff between economy of scope from product variety and economy of scale from platform sharing. Traditionally, product platform planning uses heuristic and manual approaches and relies on expertise and intuition. In this paper, we propose a data-driven method to draw the boundary of a platform, complementing other platform design approaches and assisting designers in the architecting process. The method generates a network of functions through relationships of their co-occurrences in prior designs of a product domain, and uses a network analysis algorithm to identify an optimal core-periphery structure. Functions identified in the network core co-occur cohesively and frequently with one another in prior designs, and thus are suggested for inclusion in the potential platform to be shared across a variety of product-systems with peripheral functions. We apply the method to identifying the platform functions for spherical rolling robots, based on patent data.

KEYWORDS: platform planning, function, core-periphery network

1. INTRODUCTION
Mass customization has become an important competitive edge of companies. To satisfy diverse market niches, companies develop product families based on product platforms to increase product variety with reduced cost and lead-time [1,2], in addition to other driving factors such as product flexibility [3–6]. Typically, a product family comprises a group of products that are derived from a product platform by either adding, reducing or substituting some features and functionality or scaling design parameters in the product platform [2,7–9]. A product platform is defined as a collection of common elements, components, subsystems and interfaces repeatedly used in a variety of products or systems [10–14].

A well-designed product platform is the heart of a successful product family. However, as pointed out in [15], planning or defining the product platform is one of the most challenging tasks in product family design for companies. In the previous studies, designers define a product platform mainly by identifying modules or scalable design parameters from product architectures known *a priori*. Since the architectures are typically constructed based on heuristics, expertise and intuition, it is in fact unclear how to identify which functions should be included ex ante in a platform, and analytic approaches are underdeveloped [16]. Moreover, the exiting approaches may give excessive attention to physical structures and design parameters but overlook customer needs represented by product features and functions during platform planning.

In this paper, we develop a data-driven method to draw the boundary of the potential platform by analyzing the functions and their co-occurrence relationships in the prior designs from a product domain. Particularly, the method leverages the core-periphery structure detection algorithm from the complex network analysis (or graph theory) literature to detect the optimal core-periphery partition of the function co-occurrence network into a core and periphery. The resultant network core suggests a group of functions that constitute a potential product platform, and individual or subset of the peripheral functions can be combined to generated family products.

In the following, we will review the related work, introduce the method, and then apply it to identify the potential function platform for spherical rolling robots (SRRs) based on patent data.

2. RELATED WORK

2.1 Platform Planning
Product platform planning is the key of product family design. The success of platform planning lies in the artful balance between commonality and distinctiveness [15]. In the literature,

there are two major approaches to define a product platform, namely module-based (or configurational) approach [8,17] and scale-based approach [2]. The module-based approach starts with identifying modules in product architectures [8,18]. Researchers have applied various methods and techniques to identify modules in product architectures, such as modular function deployment (MFD) [19,20], DSM [21,22], integrated DSM and MFD [23], functional modelling [24,25], clustering [26], dendrogram [27], function-component matrix [28]. Based on the identified modules, designers can develop a basic modular platform shared across a product family; from the platform, they add, remove and/or substitute one or more modules to generate product variants. Gonzalez-Zugasti, Otto and Baker [29] formulated the module-based product family architecting process as a general optimization problem to determine which modules should be made common and included in a platform.

For a scale-based approach, defining a product platform is to select design parameters that take common values, which is followed by or goes in parallel with determining the optimal values for the common and scaling design parameters [30]. Several methods have been proposed for this purpose. Some of them minimize the sensitivity of performance variations in scaling variables [2,31]; some identify common and scaling variables by analyzing the tradeoff between commonality and performance [32–34].

Most approaches reported in prior platform planning studies define platforms based on given product architectures and assume the functions or components underlying the architectures are known *a priori*. Moreover, the architectures are typically constructed through heuristic and manual approaches, relying on expertise and intuition. Therefore, it is still unclear how to draw the boundary of a platform, i.e., which functions should be included in the platform, in an analytic way. Several researchers noted this issue and proposed a few methods of platform planning based on user needs. For example, Kurtadikar and Stone [16] observed that customer needs with higher weight rated by customers and lower frequency in customer need statements constitute the core needs that lead to a base platform, based on which the functional model of the core features can be constructed for a platform design. Agard and Kusiak [35] classified customers based on their similarity in behaviors, and derived association among the requirements of a selected group of customers as a basis for functional and technical design of a platform.

In practice, functions are the intermediary through which customer needs and product features are translated into design parameters [36]. To define a product platform, designers need to outline the core functions that will be shared across variants. Thus, our method identifies core functions that have cohesively and frequently occurred together with one another and a variety of other functions in prior designs. To do so, we detect the core-periphery structure in the function co-occurrence network to reveal the core functions to inform platform planning.

2.2 Core-periphery Structure and Models

In a network exhibiting a core-periphery structure, the core refers to a set of nodes that are at the center and cohesively connected to each other, while the periphery refers to a set of nodes that are around the center and loosely connected to the core [37–39]. Since introduced in the 1950s, several types of core-periphery network structures have been observed and studied, such as rich-clubs [40,41], nested network structure [42–44], bow-tie network structure [45–47] and onion network structure [48–50].

The core and the periphery play different roles in the functioning of a complex system. For example, in social networks, players in the core take more important social roles and are more influential than those in the periphery in a community [39,51]. In cellular and flow-type networks, the core is the more degenerate and robust part with lower variability and evolvability [52]. Core-periphery network structure has also been studied in the engineering field, such as air transportation [53] and power grids [54]. The core-periphery network structure also provides a conceptual lens to analyzing product platform and family.

There are two main types of models for detecting the core-periphery structure in networks: discrete and continuous models. The discrete models simply partition the network nodes into a core class and a periphery class by maximizing a *core-periphery coefficient*, which characterize the structural distinction between the two classes. Borgatti and Everett [39] operationalized the core-periphery coefficient as the correlation between the adjacency matrices of the empirical network and an ideal core-periphery network as benchmark. In their ideal benchmark network, nodes in the core are connected to all others and the nodes in periphery are only connected to the core. They also proposed alternative benchmarks that emphasize various ideal structures and address directed and undirected, reflexive and non-reflexive netwroks, and so forth.

Besides maximizing the correlation, alternative algorithms were proposed to minimize the Hamming distance, or maximize density in the core and minimize density in the periphery [39]. In addition, Holme [55] calculated a network's core-periphery coefficient as the difference in the relative closeness centrality [39] of a maximal connected subgraph (i.e., k-core) [56] between the empirical network and the ensemble of networks with the same link sequence. To find the optimal core-periphery partition, simulated annealing [57,58], Tabu search [59], differential evolution [58,60], and genetic algorithm [61] have been used to solve such combinatorial optimization problems.

For networks that by nature do not have a clear core-periphery structure, the continuous models are more suitable, which assign a coreness score to each node rather than a dichotomous classification [39]. The coreness scores are determined by finding a vector such that the product of the vector and its transpose (i.e., the pattern matrix) is as close as possible to the adjacency matrix. Elements constituting the vector are coreness scores of corresponding nodes. Based on various measures of closeness between the two matrices, a few algorithms were proposed to optimize the correlation

coefficient, squared difference and residual between them, respectively [39,62]. The standard Fletcher-Powell [63] function maximization procedure can be used for these optimization problems. In addition, Rombach *et al.* [64] proposed to calculate a vector according to the desired core size and the sharpness of boundary between the core and the periphery of the network and then to find the optimal vector by shuffling the obtained vector to maximize the correlation between the adjacent matrix and the pattern matrix.

The discrete and continuous models have merits and limitations respectively. The discrete models offer straightforward stopping criteria to determinate a specific partition. However, the dichotomous partition might be over-simplified and arbitrary, when the network by nature does not embeded a core-periphery structure or a sharp boundary between the core and the periphery. The continuous models give a coreness score to each node, which provides more reference to weigh the nodes and adjust the core-periphery division based on other considerations. But the partition is heuristic and not deterministic.

3. OUR METHOD

In this section, we introduce a data-driven method to identify core functions for platform design. The method is based on a core-periphery analysis on the function co-occurrence network, which is constructed using empirical data of functions extracted from prior designs in a product domain. The method includes two steps.

Step 1: Construct Function Co-occurrence Network Based on Prior Designs in a Product Domain

For this step, data on prior designs in a product domain of interest is required. In the function co-occurrence network, each node is a function that appeared in any prior design; the link weight indicates the frequency a pair of functions has co-occurred in prior designs. Herein we focus on the primary functions that explain why a product exists and embody the value of a product to customers, instead of the secondary functions that realize primary functions at the lower level. In our later case study, we utilize patent data in a product domain, whereas one can also retrieve and analyze functions from other types or sources of data on prior designs.

Step 2: Detect Optimal Core-periphery Network Partition to Identify Core functions for Inclusion in a Platform

Because the function co-occurrence network is undirected and non-reflexive, we use the most common core-periphery benchmark network (i.e., 1s for core-core and core-periphery links but 0s for periphery-periphery links), and the most common model (i.e., the correlation model), to find an optimal core-periphery partition. The optimization function is given by Equation 1 together with constraints in Equation 2 for discrete model and Equation 3 for continuous model [39].

$$\text{Max } \rho = \frac{\sum_{1\le i\le n}\sum_{i<j\le n}(a_{ij}-\overline{a_{ij}})\cdot(\delta_{ij}-\overline{\delta_{ij}})}{\sqrt{\sum_{1\le i\le n}\sum_{i<j\le n}(a_{ij}-\overline{a_{ij}})^2\cdot\sum_{1\le i\le n}\sum_{i<j\le n}(\delta_{ij}-\overline{\delta_{ij}})^2}} \quad \text{(Eq. 1)}$$

$$\delta_{ij} = \begin{cases}1 \text{ if node i or node j is in the core}\\ 0 \text{ otherwise}\end{cases} \quad \text{(Eq. 2)}$$

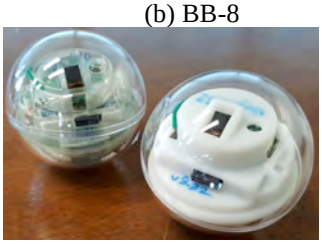
$$\delta_{ij} = c_i c_j \quad \text{(Eq. 3)}$$

where a_{ij} indicates the link weight between functions *i* and *j* (the co-occurrence frequency between them) in the function co-occurrence network, and c_i is the coreness score of function *i*. Equation 1 maximizes the Pearson correlation coefficient between the adjacent matrices of the empirical function co-occurrence network and the benchmark network. If we constrain c_i to 0s and 1s, the Equation 2 and 3 are the same. In the case study below, we employed the software UciNet [65], which uses the genetic algorithm [61] and the Fletcher-Powell [63] function maximization procedure, to solve the optimization problems for the discrete and continuous models, respectively.

4. CASE STUDY OF SPHERICAL ROLLING ROBOTS

In this section, we demonstrate the proposed method via a case study of spherical rolling robots (SRRs) and discuss the product family of Sphero, Inc. as an example of SRR platform design. SRRs are spherical shape robots that can propel themselves to roll around on the ground. Typically, the movement of SRRs is driven by a cart [66,67] or a pendulum system [68,69] contained in the shell, or other mechanisms [70–73,74,75]. SRRs have a great potential for utility in defense and consumer markets, with increased functionality and capability [76–81]. Fig. 1 presents four SRR designs. The first three are the product family developed by Sphero, Inc., and the last one is Virgo developed at Singapore University of Technology and Design (SUTD) with extreme intelligence, surveillance and reconnaissance (ISR) and autonomous capabilities.

(a) Sphero (b) BB-8 (c) BB-9E



(d) Virgo

Figure 1. (a-c) The spherical rolling robot product family from Sphero, Inc. (https://www.sphero.com/) and (d) Virgo from SUTD (https://www.sutd.edu.sg)

Step 1: Construct Function Co-occurrence Network

We analyzed a patent set of 153 SRR patents from an exhaustive SRR patent search from 1790 to May 31, 2016 [82] to extract function data. We identified the primary functions that co-occur in each SRR design described in each patent document. Table 1 lists the full set of 87 functions that we identified from the prior SRR designs disclosed in the patent set, as well as their occurrences in the patents (i.e., the count of SRR patents in which a function is identified). Such occurrences indicate the popularity of a function in the prior SRR designs. Herein, only the primary functions are identified and analyzed. For example, the function "Roll on Ground" can be decomposed into a few secondary or more generic functions, such as transmit energy, change speed and transmit motion, which are not identified and analyzed.

Table 1. The full list of functions identified from SRR patents

Function	Occurrence	Coreness (rank)	Function	Occurrence	Coreness (rank)
Roll on Ground	150	0.391(1)	Actuate Accessory	8	0.027(42)
Convert Energy	150	0.391(1)	Appoint Movement	7	0.02(46)
Store Energy	150	0.39(3)	Fly	7	0.015(49)
Control Direction	91	0.293(4)	Transform	7	0.02(46)
Remote Control	67	0.239(5)	Harvest Natural Energy	6	0.018(48)
Process Data	45	0.186(6)	Increase Movability in Water	6	0.012(50)
Store Data	42	0.182(7)	Move Eccentrically	6	0.008(52)
Self Balance	39	0.14(11)	Switch on/off Automatically	5	0.008(52)
Transmit Data	38	0.17(8)	Stabilize	4	0.01(51)
Plan Path	37	0.163(9)	Traverse Chasm	4	0.008(52)
Photograph Surrounding	35	0.153(10)	Increase Adhesive Force	4	0.007(56)
Propel in Water	31	0.083(28)	Orient Camera	3	0.007(56)
Detect Acceleration	29	0.135(12)	Thermal Control	3	0.008(52)
Measure Environment	28	0.135(12)	Resist Blast	2	0.003(65)
Track Position	27	0.13(14)	Clear Mine	2	0.003(65)
Produce Sound	27	0.111(16)	Dispose Payload	2	0.005(60)
Detect Angular Velocity	26	0.127(15)	Dock Itself	2	0.007(56)
Resist Water	26	0.074(34)	Mark Environment	2	0.003(65)
Detect Inner Part Orientation	22	0.109(17)	Passenger Control	2	0.002(69)
Emit Light	21	0.097(19)	*Follow Path*	*2*	*0.007(56)*
Communicate Information	19	0.103(18)	Track Target	2	0.005(60)
Locate Target	*18*	*0.094(20)*	Fold Itself	2	0.004(62)
Display Image/Data	18	0.091(23)	Sense Weight	2	0.002(69)
Map Surrounding Space	18	0.094(20)	Board Easily	2	0.004(62)
Avoid Obstacle	*17*	*0.044(35)*	Aim Actuator	1	0.001(76)
Play Music	17	0.085(25)	Give out Smell	1	0(85)
Sense Orientation	17	0.093(22)	Recode Angular Position	1	0.002(69)
Transport Passenger	17	0.037(38)	Obtain Annular View	1	0.001(76)
Detect Energy Level	15	0.086(24)	Park on Slope	1	0.002(69)
Record Time	15	0.085(25)	Climb	1	0.001(76)
Wireless Charge	*15*	*0.085(25)*	Detect Metal	1	0(85)
Shake/Vibrate	14	0.077(29)	Fire	1	0.001(76)
Detect Collision	13	0.077(29)	Sense Ground Contact	1	0.001(76)
Control by Gesture/Voice	13	0.075(31)	Integrate Optical Signal	1	0.001(76)
Dampen Shock	13	0.041(36)	Pick up Litter	1	0.001(76)
Modulate Light	13	0.075(31)	Pivot External Accessory	1	0.004(62)
Detect Status	13	0.075(31)	Process Image	1	0.002(69)
Traverse Obstacle	13	0.04(37)	Recover Braking Energy	1	0.001(76)
Hold External Accessory	12	0.03(40)	Dispose Chemical Pollutant	1	0.003(65)
Carry Payload	11	0.027(42)	Guarantee Safety	1	0(85)
Jump/Hop/Bounce	10	0.025(44)	Sense Pressure	1	0.002(69)
Brake	9	0.021(45)	Detect Velocity	1	0.001(76)
Detect Inclination	9	0.03(40)	Assist in Therapy	1	0.002(69)
Detect Obstacle	*9*	*0.035(39)*			

Fig. 2 visualizes the function co-occurrence network. Each node represents one of the 87 functions listed in Table 1. The node size corresponds to the occurrence of the function. The link weight indicates how frequently the corresponding pair of nodes co-occurs in prior SRR designs. In the network, functions frequently co-occurring with others, i.e., nodes having more strong links, form a cohesively connected core. This group of functions in the network core are more likely to jointly appear with varied combinations of functions in the periphery in a high number of prior SRR designs, and should be considered for inclusion in the potential SRR platform.

Step 2: Detect Optimal Core-periphery Partition

The method described in section 3 was applied to find the optimal[1] core-periphery partition of the function co-occurrence

[1] Since the genetic algorithm does not perform an exhaustive search, it may converge towards local optima rather than global optimum in some cases.

network. The resultant core includes 15 functions, which are underlined in Table 1 and highlighted as solid nodes in Fig. 2. The rest are peripheral functions and circle nodes in the outskirt of the network in Fig. 2. The corresponding core-periphery coefficient to the optimal partition is 0.528, indicating a strong but far from perfect fit with the ideal benchmark. A robustness test using Hamming distance between the empirical and benchmark network gives a similar partition.

Moreover, this empirical core-periphery coefficient value is significantly higher than the average core-periphery coefficient (0.143) of an ensemble of 100 comparable random networks generated by shuffling links in the function co-occurrence network. The z-score, i.e., (empirical value – average value of the random assemble) / standard deviation of the random assemble, is 16.1934, which indicates that the relatively high value of core-periphery coefficient is significant at 5% level. That is, the empirical function co-occurrence network presents a core-periphery structure to a significant degree. And the discrete model well captures its structure.

More detailed descriptions of the 15 core functions are provided in Table 2, together with choices of components to fulfill these functions. Among the 15 core functions, Roll on Ground, Convert Energy, Store Energy, Control Direction and Self Balance are required to propel a robot, while the others are generally related to controlling a robot and monitoring the environment. These functions constitute the basic functionality of a general autonomy and sensing mobile robot, and thus should be included in a platform. Then a family of SRR products for specific function purposes can be developed via integrating some peripheral functions with the platform.

For example, designers may design an autonomous SRR by integrating some peripheral functions, such as Detect Obstacles, Avoid Obstacles, Locate Target, Map Surrounding Space, and Follow Path, with the platform functions. To develop recreational SRRs, designers can consider such peripheral functions, such as Play Music, Emit Light, Modulate Light and Shake/Vibrate, on top of the platform functions. Designers may make different choices to combine the peripheral functions with the platform ones to yield a family of SRR product family, based on their expertise or customer needs.

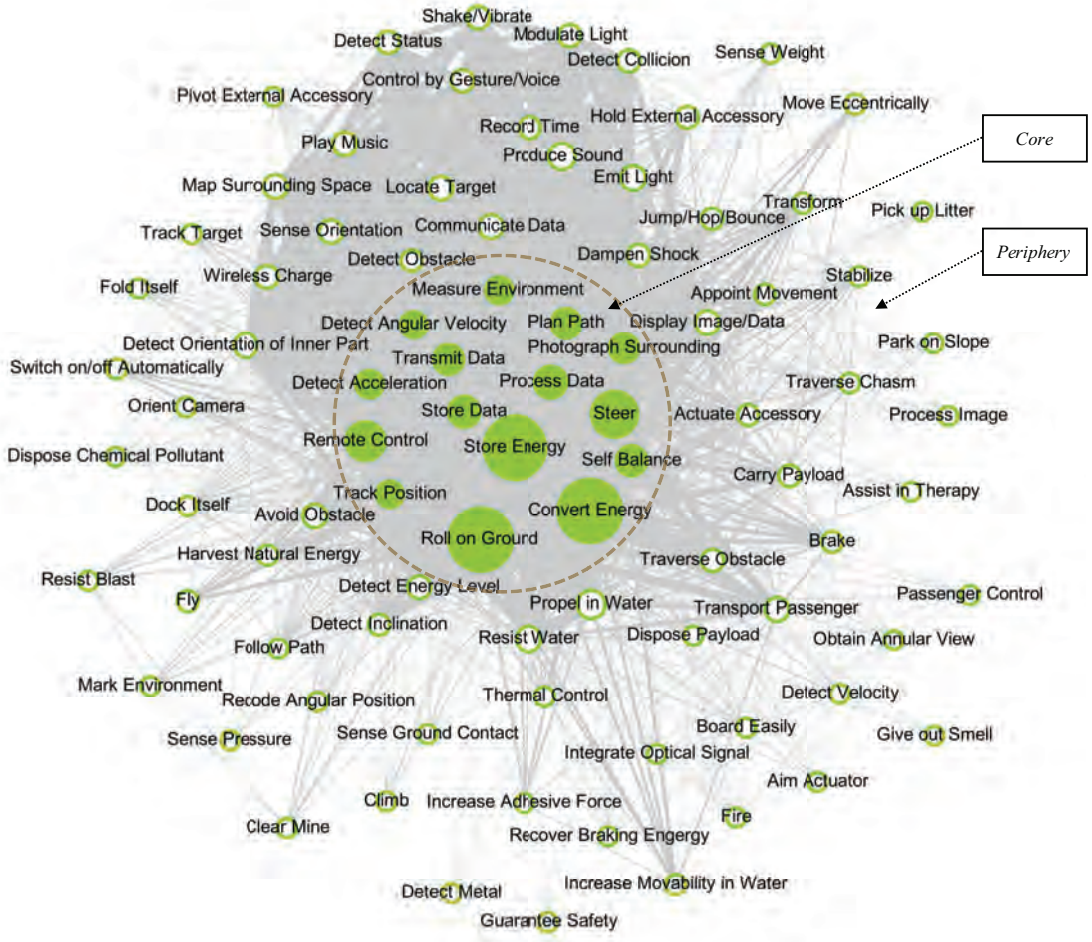


Figure 2. The co-occurrence network of functions identified in prior SRR designs. Core functions are highlighted as solids and periphery functions are presented as circles.

Table 2. The list of core functions for a SRR platform design

Core Functions	Description	Component Choice
Roll on Ground	Roll on ground in a self-propelled manner.	Drive System, Wheels
Covert Energy	Convert stored energy to kinetic energy for driving.	Electric Motors
Store Energy	Store Energy.	Battery
Control Direction	Control the driving direction of the robot.	Propelled Wheels
Remote Control	Remotely controlled by an external controller.	Bluetooth
Transmit Data	Send data to controller or display devices.	Bluetooth
Process Data	Process data for controlling the movement of the robot and fulfilling other functions.	Micro-controller Board
Store Data	Store data, such images, sounds, etc..	Memory
Self Balance	Keep Itself in a ready position for driving. For example, avoid the inner cart turning upside down	Supporting Pod
Plan Path	Plan a path to get a given target.	Micro-controller Board
Photograph Surrounding	Generate images of the surroundings of the robot.	Camera
Detect Acceleration	Detect the acceleration of an inner part of the robot.	Accelerometer (IMU)
Detect Angular Velocity	Detect the angular velocity of an inner part of the robot.	Gyroscope (IMU)
Measure Environment	Measure the surrounding environment of the robot, such as temperature, light intensity, radioactivity and nuclear).	Sensors for Measuring Temperature, etc.
Track Positions	Track the absolute positions of the robot.	GPS

Particularly, the first 9 platform functions listed in Table 2 constitute most of the functions of a popular SRR product in the market, the Sphero robot toy (Fig. 1a) from Sphero, Inc. The Sphero robot toy also includes some other functions, such as Emit Light and Wireless Charge. The company's later released BB-8 model (Fig. 1b) included more functions, such as Voice Control, Follow Path, Produce Sound (for sound effect), Hold & Pivot External Accessory (a rotating head) to mimic a robot in the Star Wars movie, and the BB-9E model (Fig.1c) further included an AR function to BB-8. Meanwhile, the first 13 core functions listed in Table 2 appear in Virgo (Fig. 1d), a platform robot developed in SUTD, on top of which more functions will be integrated for defense or entertainment. In addition to the 13 functions, Virgo also has such functions as Locate Target, Follow Path, Detect Obstacle, and Avoid Obstacle.

In practice, designers should take the identified core functions identified from the data-driven algorithm as a reference point, rather than a rigid result. In the case of the SRRs, the designers or companies may include additional functions than the 15 core ones or a subset of them in a platform, by taking into account user and market needs, their own technical expertise, as well as the company's strategy. The designers may also examine the coreness scores or other measures of individual functions to decide whether to include it in the platform. The coreness score of each function resulting from the continuous model with its rank by the score in parentheses is also listed in Table 1. The core functions identified by the discrete model have the highest coreness scores. In Table 1, we italicize the functions appearing in Virgo and Sphero's product platform but not included in the core. Most of them have relatively high coreness scores as well. The coreness scores may provide another reference to weigh the functions for inclusion in a product platform for designers.

5. DISCUSSION

In this paper, we develop a data-driven method to identify the functions to be considered for inclusion into a product platform. The method applies core-periphery network analysis to a function network constructed based on the relationship of function co-occurrences in prior designs in a product domain. In the case study of SRRs, 15 potential platform functions are identified using the network based data-driven method. As suggested in Table 1, these core functions are among the most frequently occurring ones in prior design, but not always. For example, the function, Propel in Water, has a high occurrence (ranking at the 8th) but is founded not in the network core. In particular, our method provides an algorithm-based stopping point in the search for functions to include in a platform. The resultant dichotomous partition of core and peripheral functions may serve as a reference point, based on which designers may add some peripheral functions or remove some core functions for an eventual platform design according to their expertise, customer needs and specific strategies. Apart from the discrete core-periphery partition, the coreness scores from a continuous model also provide additional reference for designers to make an adjustment and a final decision.

In our case study, we included all the related patents in the analysis and treated all of them the same. In practice, designers may classify the prior designs and then identify the platform functions for different market niches. For example, designers may analyze only the SRR patents classified in the international patent class (IPC) A63 "Sports and Entertainment" to develop a platform for toys, or analyze the SRR patents classified in B62 "Land Vehicles" to design a platform for a family of SRR products functioning as vehicles. In addition, designers can consider and select functions in the periphery of the function co-occurrence network to generate variants. That is, combining

different subsets of peripheral functions with the fixed set of platform functions results in a product family.

Meanwhile, functions identified from products have been used for long to abstract design tasks and model product architectures and product family architectures [83–85]. The data-driven method for identifying platform functions can serve as a prior step before identifying modules from a product architecture through function-based approaches, such as MFD approach [19,20], functional-model approach [25,86], function-component-matrix approach [28] and dendrogram approach [27]. Specifically, the identified platform functions can be further decomposed into secondary functions to inform the design of modules within the platform.

The proposed method also has limitations. The first is that rich data on prior designs in the product domains are required. In this study, we only demonstrate the use of patent data. Other forms of data on prior designs in a field should be explored as well. In addition, we only present the case study of SRRs in this paper. In future research, we anticipate applying this method to other platform design cases. We may get better understanding of the contextual factors that influence the choice of core-periphery models and the effectiveness of the data-driven method through the broader applications.

5. CONCLUDING REMARKS

In this paper, we present a data-driven method to identify the functions for inclusion in a potential product platform. The method is centered on detecting the optimal core-periphery partition in the function co-occurrence network, which is constructed based on the function data in prior designs of a product domain. We demonstrate the method by applying it to identify functions in the network core to inform the platform design of spherical rolling robots based on the relevant patent data, and to forming product family by integrating the platform functions with varied sets of functions in the periphery of the function co-occurrence network. In brief, the data-driven method assists in product platform and family design, and complements the existing heuristic and qualitative approaches of platform planning.

ACKNOWLEDGMENTS

This research was funded in part by the SUTD-MIT International Design Centre (IDC, idc.sutd.edu.sg) and Singapore Ministry of Education Academic Research Fund Tier 2.

REFERENCES

[1] J. B. Pine, *Mass customization: the new frontier in business competition*. Boston: Harvard Business School Press, 1993.

[2] T. W. Simpson, C. C. Seepersad, and F. Mistree, “Balancing commonality and performance within the concurrent design of multiple products in a product family,” *Concurr. Eng. Res. Appl.*, vol. 9, no. 3, pp. 177–190, 2001.

[3] P. K. P. Rajan, M. Van Wie, M. I. Campbell, K. L. Wood, and K. N. Otto, “An empirical foundation for product flexibility,” *Des. Stud.*, vol. 26, no. 4, pp. 405–438, 2005.

[4] D. A. Keese, A. H. Tilstra, C. C. Seepersad, and K. L. Wood, “Empirically-Derived Principles for Designing Products With Flexibility for Future Evolution,” *Vol. 3 19th Int. Conf. Des. Theory Methodol. 1st Int. Conf. Micro- Nanosyst. 9th Int. Conf. Adv. Veh. Tire Technol. Parts A B*, no. January, pp. 483–498, 2007.

[5] D. a. Keese, C. C. Seepersad, and K. L. Wood, “Product flexibility measurement with enhanced Change Modes and Effects Analysis (CMEA),” *Int. J. Mass Cust.*, vol. 3, no. 2, p. 115, 2009.

[6] M.-A. Cardin, “Enabling Flexibility in Engineering Systems: A Taxonomy of Procedures and a Design Framework,” *J. Mech. Des.*, vol. 136, no. 1, p. 11005, 2013.

[7] T. Simpson, “Product platform design and optimization: status and promise,” *ASME 2003 Int. Des. …*, pp. 3–20, 2003.

[8] K. Ulrich, “The role of product architecture in the manufacturing firm,” *Res. Policy*, vol. 24, no. 3, pp. 419–440, 1995.

[9] M. H. Meyer and J. M. Utterback, “The product family and the dynamics of core capability,” *Sloan Manage. Rev.*, vol. 34, no. 3, pp. 29–47, 1993.

[10] B. Wilhelm, “Platform and modular concepts at Volkswagen—their effects on the assembly process,” in *Transforming automobile assembly*, Berlin, Heidelberg: Springer, 1997, pp. 146–156.

[11] M. H. Meyer and A. P. Lehnerd, *The power of product platforms: building value and cost leadership*. New York: Business & Economics, 1997.

[12] M. E. McGrath, *Product strategy for high-technology companies: how to achieve growth, competitive advantage, and increased profits*. Irwin Professional Pub., 1995.

[13] D. Robertson and K. Ulrich, “Planning for Product Platforms,” *Sloan Manag. Rev.*, vol. 39, no. 4, pp. 19–31, 1998.

[14] G. Ericsson, Anna; Erixon, *Controlling design variants: modular product platforms*. Society of Manufacturing Engineers, 1999.

[15] T. W. Simpson, Z. Siddique, and J. Jiao, *Product platform and product family design: Methods and applications*. 2006.

[16] R. M. Kurtadikar and R. B. Stone, “Investigation of customer needs frequency vs. weight in product platform planning,” *Proc. IMECHE 2003 ASME Int. Mech. Eng. Congr. R D Expo*, pp. 375--387, 2003.

[17] X. Du, J. Jiao, and M. M. Tseng, “Architecture of Product Family: Fundamentals and Methodology,” *Concurr. Eng.*, vol. 9, no. 4, pp. 309–325, 2001.

[18] K. T. Ulrich and K. Tung, “Fundamentals of Product Modularity,” in *Proceedings of the 1991 Winter Annual Meeting, DE- Vol. 39, Atlanta, GA.*, 1991.

[19] A. Erlandsson, G. Erixon, and B. Ostgren, “Product modules-the link between QFD and DFA,” in *The International Forum on Product Design for Manufacture and Assembly, Newport, RI.*, 1992.

[20] G. Erixon and B. Ostgren, “Synthesis and evaluation tool for modular designs,” in *International conference on engineering design*, 1993, pp. 898–905.

[21] T.-L. Yu, A. A. Yassine, and D. E. Goldberg, “An information theoretic method for developing modular architectures using genetic algorithms,” *Res. Eng. Des.*, vol. 18, no. 2, pp. 91–109, 2007.

[22] T.-L. Yu, A. A. Yassine, and D. E. Goldberg, “A genetic algorithm for developing modular product architectures,” in *ASME 2003 international design engineering technical conferences and computers and information in engineering conference*, 2003, pp. 515–524.

[23] J. Malmström and J. Malmqvist, “Trade off analysis in product structures: A case study at Celsius Aerotech,” in *Proceedings of NordDesign’98*, 1998, pp. 187–196.

[24] R. B. Stone, K. L. Wood, and R. H. Crawford, “A Heuristic Method for Identifying Modules for Product Architectures,” *Des. Stud.*, vol. 21, pp. 1–47, 2000.

[25] J. B. Dahmus, J. P. Gonzalez-Zugasti, and K. N. Otto, “Modular product architecture,” *Des. Stud.*, vol. 22, no. 5, pp. 409–424, 2001.

[26] S. M. Salhieh and A. K. Kamrani, “Macro level product development using design for modularity,” *Robot. Comput. Integr. Manuf.*, vol. 15, no. 4, pp. 319–329, 1999.

[27] K. Hölttä, V. Tang, and W. P. Seering, “Modularizing product architectures using dendrograms,” in *DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design*, 2003.

[28] S. K. Fixson, “Product architecture assessment: A tool to link product, process, and supply chain design decisions,” *J. Oper. Manag.*, vol. 23, no. 3–4, pp. 345–369, 2005.

[29] J. P. Gonzalez-Zugasti, K. N. Otto, and J. D. Baker, “Method for architecting product platforms,” *Res. Eng. Des. - Theory, Appl. Concurr. Eng.*, vol. 12, no. 2, pp. 61–72, 2000.

[30] T. W. Simpson, “Product platform design and customization: Status and promise,” *Ai Edam*, vol. 18, no. 1, pp. 3–20, 2004.

[31] A. Messac, M. P. Martinez, and T. W. Simpson, “Effective product family design using physical programming,” *Eng. Optim.*, vol. 34, no. 3, pp. 245–261, 2002.

[32] R. U. Nayak, W. Chen, and T. W. Simpson, “A variation-based method for product family design,” *Eng. Optim.*, vol. 34, no. 1, pp. 65–81, 2002.

[33] A. Messac, M. P. Martinez, and T. W. Simpson, “A penalty function for product family design using physical programming,” *ASME J. Mech. Des.*, vol. 124, no. 2, pp. 164–172, 2002.

[34] R. Fellini, M. Kokkolaras, P. Y. Papalambros, and A. Perez-Duarte, “Platform selection under performance loss constraints in optimal design of product families,” *ASME 2002 Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf.*, no. 734, pp. 613–621, 2002.

[35] B. Agard and A. Kusiak, “Data-mining-based methodology for the design of product families,” *Int. J. Prod. Res.*, vol. 42, no. 15, pp. 2955–2969, 2004.

[36] J. Jiao, T. W. Simpson, and Z. Siddique, “Product family design and platform-based product development: A state-of-the-art review,” *J. Intell. Manuf.*, vol. 18, no. 1, pp. 5–29, 2007.

[37] P. Csermely, A. London, L.-Y. Wu, and B. Uzzi, “Structure and dynamics of core/periphery networks,” *J. Complex Networks*, vol. 1, no. 2, pp. 93–123, 2013.

[38] B. Yan and J. Luo, “Multicore-Periphery Structure in Networks,” 2016.

[39] S. P. Borgatti and M. G. Everett, “Models of corer/periphery structures,” *Soc. Networks*, vol. 21, pp. 375–395, 1999.

[40] V. Colizza, A. Flammini, M. A. Serrano, and A. Vespignani, “Detecting rich-club ordering in complex networks,” *Nature*, vol. 2, no. February, pp. 110–115, 2006.

[41] S. Zhou and R. J. Mondragon, “The Rich-Club Phenomenon In The Internet Topology,” *IEEE Commun. Lett.*, vol. 8, no. 3, pp. 180–182, 2004.

[42] S. Saavedra, F. Reed-Tsochas, and B. Uzzi, “A simple model of bipartite cooperation for ecological and organizational networks,” *Nature*, vol. 457, no. 7228, p. 463, 2009.

[43] S. Saavedra, D. B. Stouffer, B. Uzzi, and J. Bascompte, “Strong contributors to network persistence are the most vulnerable to extinction,” *Nature*, vol. 478, no. 7368, p. 233, 2011.

[44] S. Bustos, C. Gomez, R. Hausmann, and C. A. Hidalgo, “The Dynamics of Nestedness Predicts the Evolution of Industrial Ecosystems,” *PLoS One*, vol. 7, no. 11, p. e49393, 2012.

[45] J. Supper, L. Spangenberg, H. Planatscher, A. Dr??ger, A. Schr??der, and A. Zell, “BowTieBuilder: Modeling signal transduction pathways,” *BMC Syst. Biol.*, vol. 3, pp. 1–13, 2009.

[46] A. Broder *et al.*, “Graph structure in the web,” *Comput. Networks*, vol. 33, no. 1–6, pp. 309–320, 2000.

[47] H. Kitano and K. Oda, “Robustness trade-offs and host-microbial symbiosis in the immune system,” *Mol. Syst. Biol.*, vol. 2, pp. 1–10, 2006.

[48] H. J. Herrmann, C. M. Schneider, A. A. Moreira, J. S. Andrade, and S. Havlin, “Onion-like network topology enhances robustness against malicious attacks,” *J. Stat. Mech. Theory Exp.*, vol. 2011, no. 1, pp. 1–9, 2011.

[49] E. S. Division, P. Program, and C. De Azurem, “Engineering Design and Product Development : a focus of the MIT-Portugal Programme *,” vol. 24, no. 2, pp. 336–344, 2008.

[50] C. M. Schneider, A. A. Moreira, J. S. Andrade, S. Havlin, and H. J. Herrmann, “Mitigation of malicious attacks on network observation,” *Proc. Natl. Acad. Sci.*, vol. 108, no. 10, pp. 3838–3841, 2011.

[51] B. Collier and R. Kraut, “Leading the collective: Social capital and the development of leaders in core-periphery organizations,” 2012.

[52] P. Tieri *et al.*, “Network, degeneracy and bow tie integrating paradigms and architectures to grasp the complexity of the immune system,” *Theor. Biol. Med. Model.*, vol. 7, no. 1, pp. 1–16, 2010.

[53] T. Opsahl, V. Colizza, P. Panzarasa, and J. J. Ramasco, “Prominence and Control: The Weighted Rich-Club Effect,” *Phys. Rev. Lett.*, vol. 101, no. 16, p. 168702, 2008.

[54] J. J. McAuley, L. Da Fontoura Costa, and T. S. Caetano, “Rich-club phenomenon across complex network hierarchies,” *Appl. Phys. Lett.*, vol. 91, no. 8, pp. 2–5, 2007.

[55] P. Holme, “Core-periphery organization of complex networks,” *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 72, no. 4, 2005.

[56] S. B. Seidman, “Network structure and minimum degree,” *Soc. Networks*, vol. 5, no. 3, pp. 269–287, 1983.

[57] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, “Optimization by Simulated Annealing,” *Science (80-.).*, vol. 220, no. 4598, pp. 671–680, 1983.

[58] J. Boyd, W. Fitzgerald, and R. J. Beck, “Computing core / periphery structures and permutation tests for social relations data,” *Soc. Networks*, no. 28, pp. 165–178, 2007.

[59] F. Glover, “Tabu Search - Part I,” *ORSA J. Comput.*, vol. 2 1, no. 3, pp. 4–32, 1989.

[60] R. Storn and K. Price, “Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces,” *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[61] D. E. Goldberg, *Genetic algorithms*. Pearson Education India, 2006.

[62] A. L. Comrey, “The Minimum Residual Method of Factor Analysis,” *Psychol. Rep.*, vol. 11, no. 1, pp. 15–18, 1962.

[63] W. H. Press, *Numerical recipes in Pascal: the art of scientific computing*. Cambridge University Press, 1989.

[64] M. P. Rombach, M. A. Porter, J. H. Fowler, and P. J. Mucha, “Core-Periphery Structure in Networks,” pp. 1–27, 2012.

[65] S. P. Borgatti, M. G. Everett, and L. . Freeman, “Ucinet for Windows: Software for Social Network Analysis,” *Harvard, MA: Analytic Technologies*, 2002. .

[66] a. Bicchi, a. Balluchi, D. Prattichizzo, and a. Gorelli, “Introducing the “SPHERICLE”: an experimental testbed\nfor research and teaching in nonholonomy,” *Proceedings of International Conference on Robotics and Automation*, vol. 3. 1997.

[67] I. H. Bernstein and A. Wilson, “Self-propelled device with actively engaged drive system,” U.S. Patent 9,766,620, 2017.

[68] J. Kim, H. Kwon, and J. Lee, “A rolling robot: design and implementation,” *Asian Control Conf. 2009. ASCC 2009. 7th*, pp. 1474–1479, 2009.

[69] J. C. Yoon, S. S. Ahn, and Y. J. Lee, “Spherical robot with new type of two-pendulum driving mechanism,” *INES 2011 - 15th Int. Conf. Intell. Eng. Syst. Proc.*, vol. 1, pp. 275–279, 2011.

[70] S. Bhattacharya and S. K. Agrawal, “Design, experiments and motion planning of a spherical rolling robot,” *Robot. Autom. 2000. Proceedings. ICRA’00. IEEE Int. Conf.*, vol. 2, no. April, pp. 1207–1212, 2000.

[71] S. Mukherjea, B. Bamba, and P. Kankar, “Information Retrieval and Knowledge Discovery Utilizing a BioMedical Patent Semantic Web,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 8, pp. 1099–1110, 2005.

[72] G. A. Hajos, J. A. Jones, A. Behar, and M. Dodd, “An Overview of Wind-Driven Rovers for Planetary Exploration,” pp. 1–13, 2018.

[73] A. Halme, J. Suomela, T. Schönberg, and Y. Wang, “a Spherical Mobile Micro-Robot for,” in *ESA Workshop on Advanced Space Technologies for Robot Applications*, 1996, no. April, pp. 1–3.

[74] T. Nemoto, R. E. Mohan, and M. Iwase, “Realization of rolling locomotion by a wheel-spider-inspired hexapod robot,” *Robot. Biomimetics*, vol. 2, no. 1, p. 3, 2015.

[75] T. Yanagida, R. E. Mohan, K. Pathmakumar, Thejus Elangovan, and M. Iwase, “Design and Implementation of a Shape Shifting Rolling–Crawling–Wall-Climbing Robot,” *Appl. Sci.*, vol. 7, no. 4, p. 342, 2017.

[76] F. Wu, L. Marechal, A. Vibhute, S. Foong, G. S. Soh, and K. L. Wood, “A compact magnetic directional proximity sensor for spherical robots,” in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 2016, pp. 1258–1264.

[77] F. Wu, A. Vibhute, G. S. Soh, K. L. Wood, and S. Foong, “A compact magnetic field-based obstacle detection and avoidance system for miniature spherical robots,” *Sensors (Switzerland)*, vol. 17, no. 6, 2017.

[78] X. Niu, A. P. Suherlan, G. S. Soh, S. Foong, K. Wood, and K. Otto, “Mechanical development and control of a miniature nonholonomic spherical rolling robot,” in *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, 2014, pp. 1923–1928.

[79] V. A. Ajay, A. P. Suherlan, G. S. Soh, S. Foong, K. Wood, and K. Otto, “Detc2015-47223 Localization and Trajectory Tracking of an Autonomous,” pp. 1–9, 2015.

[80] A. R. Chowdhury, A. Vibhute, G. S. Soh, S. H. Foong, and K. L. Wood, “Implementing caterpillar inspired roll control of a spherical robot,” *Proc. - IEEE Int. Conf. Robot. Autom.*, no. September, pp. 4167–4174, 2017.

[81] A. R. Chowdhury *et al.*, “Experiments in Second Order Sliding Mode Control of a CPG based Spherical Robot,” pp. 2401–2408, 2017.

[82] B. Song and J. Luo, “Mining Patent Precedents for Data-Driven Design : The Case of Spherical Rolling Robots,” *J. Mech. Des.*, vol. 139, no. 11, p. 111420, 2017.

[83] F. Erens and K. Verhulst, “Architectures for product families,” *Comput. Ind.*, vol. 33, no. 2–3, pp. 165–178, 1997.

[84] R. B. Stone and K. L. Wood, “Development of a Functional Basis for Design,” *J. Mech. Des.*, vol. 122, no. December 2000, pp. 359–370, 2000.

[85] J. Hirtz, R. B. Stone, D. A. McAdams, S. Szykman, and K. L. Wood, “A Functional Basis for Engineering Deisgn : Reconciling and Evolving Previous Efforts,” *Res. Eng. Des.*, vol. 13, no. 2, pp. 65–82, 2002.

[86] R. B. Stone, K. L. Wood, and R. H. Crawford, “A heuristic method for identifying modules for product architectures,” *Des. Stud.*, vol. 21, no. 1, pp. 5–31, 2000.