Volume 5, Issue 1
January 2012
ISSN: 1946-1836

JOURNAL OF INFORMATION SYSTEMS APPLIED RESEARCH

In this issue:

The **Journal of Information Systems Applied Research** (JISAR) is a double-blind peer-reviewed academic journal published by **EDSIG,** the Education Special Interest Group of AITP, the Association of Information Technology Professionals (Chicago, Illinois). Publishing frequency is currently quarterly. The first date of publication is December 1, 2008.

JISAR is published online (http://jisar.org) in connection with CONISAR, the Conference on Information Systems Applied Research, which is also double-blind peer reviewed. Our sister publication, the Proceedings of CONISAR, features all papers, panels, workshops, and presentations from the conference. (http://conisar.org)

The journal acceptance review process involves a minimum of three double-blind peer reviews, where both the reviewer is not aware of the identities of the authors and the authors are not aware of the identities of the reviewers. The initial reviews happen before the conference. At that point papers are divided into award papers (top 15%), other journal papers (top 30%), unsettled papers, and non-journal papers. The unsettled papers are subjected to a second round of blind peer review to establish whether they will be accepted to the journal or not. Those papers that are deemed of sufficient quality are accepted for publication in the JISAR journal. Currently the target acceptance rate for the journal is about 45%.

Questions should be addressed to the editor at editor@jisar.org or the publisher at publisher@jisar.org.

JOURNAL OF INFORMATION SYSTEMS APPLIED RESEARCH

Editors

JISAR Editorial Board

Co-Creating Value in Systems Development: A Shift towards Service-Dominant Logic

Jeffry S. Babb, Jr.
jbabb@wtamu.edu
Computer Information & Decision Management
West Texas A&M University
Canyon, TX 79119

Mark Keith
mjkeith@cba.ua.edu
Management Information Systems
University of Alabama
Tuscaloosa, AL 35487

Abstract

As agile systems development methods can be viewed from a disruptive technology perspective, what have we learned from the perturbation? Our perspective does not focus on how agility changed existing methods, but rather on what changes in the environment precipitated agile methods and what can be learned about the future of systems development from these changes. In this paper, we re-conceptualize systems development methods from both a service-dominant logic perspective and from the perspective of the co-creation of value between the systems developer and the customer during the systems development life cycle (SDLC). In software development, value co-creation happens in the form of meeting customer needs as well as the creation of new operant resources. We provide a new conceptualization of systems development method selection based on these ideas and illustrate some implications from both the S-DL and Co-creation perspectives. This conceptualization should afford new areas for future research which assumes that agile vs. plan-driven methodology choice is a false dichotomy.

Keywords: Systems Development Methods, Service-Dominant Logic, Co-Creation, Agile.

1. IMPORTANT INFORMATION

The question of systems methodology selection is still of primary concern and the landscape of systems development methodologies seems as confusing as ever. Increasingly, both in practice and in research, we see the "disciplining" of agile methods and the "lightening" or "agile-ization" of traditional plan-based methods such that hybridization is becoming normative. Whereas systems development method selection is typically made with an eye towards better performance, the sum of contributing factors informing method selection are still not entirely understood or agreed upon (Chow and Cao 2008). While the advent and adoption of agile methods has been well-documented (Boehm and Turner 2004), we would argue that the societal and environmental precipitations to agile and lightweight methods are not fully understood.

This paper draws from the marketing theories surrounding service-dominant logic (S-DL) which can be used to explain the disruption of "old"

systems development methods and the evolution and maturation of new "lightweight" methods which favor agility. Changes are afoot both in plan-driven and agile methods, where each are increasingly influencing the other such that orthodoxy in either methodological tradition is the exception rather than the rule. We propose that both ends of the methodological spectrum are responding to the new and emergent demands in the marketplace for co-creation between the software developer and the customer (Prahalad & Ramaswamy, 2004a, 2004b). In this sense, we include, extend, and refine the notion of *task environment* to account for a service-dominant logic and the co-creation of value between developer and customer. From this perspective, we propose that agile methods have been conceptualized under a false dichotomy – agile vs. plan-driven methods – which may have left other more important questions regarding systems development method selection unanswered.

Perhaps the question we should ask is: "what has changed in the environment?" or perhaps "what changed in the software development market?" If methodology choice has traditionally been under the project manager's purview to match the characteristics of the software development method to the constraints of the task environment (Saunders & Scammel, 1986), then something in the environment—or perhaps market—must have changed to necessitate agile methods. We posit that the actual structure of most systems development methodologies has become more fluid as hybrid methods are appearing. This trends towards fluidity isn't accidental; there is now a shift towards new principles, which have evolved in parallel in the marketing discipline (Vargo & Lusch, 2004), governing the way markets work and how value is created between a service provider and a customer. Have these changes in the structuration of markets between customer and service-provider—changes in the sense of Giddens (1984), and perhaps even Kuhn (1996)—contributed to systems development method confusion? What can systems development practitioners and researchers learn from this shift? How can we re-envision the advent of agile and what does it mean to future method selection?

Too often we use traditional cost-based or goods-based approaches to measure project success, such as the degree to which software was developed on time or on budget. However, we see a new perspective where the customer and provider co-create value and where the service provider and the customer mutually shape the meaning of value as they interact. In this sense, the software service provider and the customer are mutually constructing and co-creating value through their interactions. This fresh perspective on method selection, method effectiveness, and on the philosophies informing systems development methods stems from the work on co-creation (Prahalad & Ramaswamy, 2000, 2004a, 2004b, 2004c, 2006) and S-DL (Vargo and Lusch, 2004, 2008) which transpired, seemingly in parallel, with the advent and rise of agile methods. Their new ideas suggest that the structuring of value in systems development is no longer a tit-for-tat stepwise process between releases. Rather, value is now co-created continually between the systems development provider and the customer as they progress through the stages of the systems development life cycle (SDLC). In this sense, the co-creation and structuring of customer (and developer) value needn't wait until the project is done, structuring happens from the moment of inception.

The paper proceeds as follows. First, we present the current thinking in the background literatures on the agile vs. plan-driven methods conundrum. We also discuss how this false dichotomy influences attitudes on method selection. Next, we illustrate new thinking based on S-DL and the co-creation of value and how these concepts change our own views on systems development method selection. In the next section, we postulate a revised conceptualization for systems development methods informed by utilizing an S-DL to focus on the co-creation of value with the customer. We then proceed to illustrate the implications this new conceptualization would have in a few cases. Lastly, we next offer discussion and directions for future research.

2. CURRENT THINKING

The traditional imprint on thinking about systems development methods holds that the outputs of this endeavor are "goods," and increasingly, a service. However, what if a systems development method also, and primarily, facilitated the ongoing co-creation of value in a customer-provider relationship? Under this light, prevailing ideas on the purpose of a systems development methodology, understood on axes related to tolerance of risk

and change, and understood as a choice between plan-driven and agile methods, are a false dichotomy. Rather, it is possible to realize co-created value from a variety of systems development methods. This is so as method selection undertaken from an S-DL and co-creation perspective allows for a customer-provider co-creative team to achieve method fit and service/good personalization. The following sections outline the theories informing this assertion.

Co-creation of Value

Co-creation has arisen with the upset of traditional systems development methods, in favor of agile methods, as a response to environmental changes. Originally offered as a strategy in the marketing field, co-creation holds that a service-/solution-provider and a customer mutually construct and reconstruct value by exploring design, learning, and meaning in a shared partnership (Pralahad & Ramaswamy, 2000, 2004a; Sanders & Stappers, 2008; Payne et al. 2008). When one examines the tenets of agile software development methods, the idea at once becomes familiar. However, the association between trends towards co-creation and changes in software and systems development methods are only slowly coming to light (Kar, 2006; Madsen & Matook, 2010). What co-creation compellingly shares with the development and evolution of agile methods is a change in perspective regarding the role the customer plays in value creation.

Co-creation can be understood in the following scenario. In the days where consumer software was commonly purchased in a brick-and-mortar mode, a software developer, such as Microsoft, would use several techniques to gauge and gather customer input on desirable software features prior to the release of a new iteration of an old product, or as a facet of market analysis for a new product. In any case, the customer was the recipient of a good (the packaged software), which was completed after running a fairly traditional requirements and analysis phase of the SDLC. In this model, the customer wasn't a partner, rather the customer was a semi-passive recipient of goods after the developer had sequestered away when the requirements, analysis and design phases were complete. When the product was released, the customer either received something that was "one-size-fits-all," or was, at the very least, customizable.

Co-creation takes a different tact and has been facilitated by new avenues for customer/provider interaction over the past two decades. Most of these new avenues for interaction involved utilization of the Internet subsequent to its mass commercialization in the 1990s. The Internet, and its applications such as the World Wide Web, has allowed the customer and service-provider to co-create value in the form of unique, tailored, bespoke, and personalized services and experiences (Prahalad & Ramaswamy, 2004c). Rather more compelling, from a systems development perspective, are the payoffs of learning and the loyalty, relationships, and renown that the service-provider enjoys in the co-creation relationship. That the customer provides early signals of value and a means of learning has been well-established in the literature on agile methods (Babb, 2009; Boehm and Turner, 2004). Co-creation can explain changes in the environment related to the means by which information is disseminated: networked, ever-present, contextual, and memetic. For many of the compelling and society-changing technologies enabled by the Internet and World Wide Web – eCommerce, social networks, peer to peer - value is usually co-created with customers as the customer is able to personalize their interactions with the service-provider and the goods/services they consume. This personalization, this tailoring, is a large part of the co-creation proposition.

We see this co-creation phenomenon in agile methods and we see it in hybrid methods: the degree to which the demand for personalization – not necessarily specialization, or customization – influences customer relations and influences method selection. If agile is about managing change, then perhaps the growing acculturation to co-creative pathways of customer/provider relations is what is driving this change. We speak of Apple, Netflix, eBay, Amazon, and Facebook as being respective technology leaders in so far as they each afford the customer simple and personalized choices for interaction. Thus, if value is increasingly co-created within the provider/customer relationship, rather than being created entirely by the provider, then systems development methods which accommodate this bilateral flow will allow the service-provider to adapt and foster the myriad relationships made possible by adopting systems development methods which are co-creative in nature.

The Traditional View of the Market

The Co-creation of value and agile methods share the view that the customer is not merely a passive target for transactions, but rather that the customer is an integral part of the processes of design, planning, and strategy. As with co-creation, agile methods allow a customer representative the opportunity to craft their own product, and their own experience, by engaging the agile partnership.

The relationship between the firm and the customer in traditional markets is represented in Figure 1 (Prahalad & Ramaswamy, 2004a). In this conceptualization of a market, the firm utilizes the market in order to extract value from the customer. In turn, the customer extracts value from the receipt of goods or services. In this conceptualization, there is strong directionality from the firm, making the customers' role very lopsided and unequal. In this traditional market scenario, the firm believes it is the sole creator of value in providing optimized and undifferentiated services. Furthermore, in this mode of operation the firm is the sole source of expertise and acts as exclusive arbiter of value, optimization, and cost reduction. As a result, IT project success is viewed in terms of being on-time and on-budget. As Prahalad and Ramaswamy (2004a) put it:

As long as firms believe that the market can be separated from the value creation process, firms in search of sources of value will have no choice but to squeeze as much costs from their "value chain" activities as possible. Meanwhile, globalization, deregulation, outsourcing, and the convergence of industries and technologies are making it much harder for managers to differentiate their offerings. Products and services are facing commoditization as never before. Companies can certainly not escape being super-efficient. However, if consumers do not see any differentiation they will buy smart and cheap. The result is the "Walmartization" of everything, from clothes to DVD players.

We can see this phenomenon in traditional systems development methods as well. While well-meaning, process optimization approaches, such as the Capability Maturity Model Integrated, or the Rational Unified Process, work at systems of efficiency and cost-savings in order to produce the same type of product reliably. While these traditional plan-driven approaches provide a reasonable hedge against risk, the requirements process tends to force customer needs into templates, such as those provided by the UML and by the CASE tools that support them. This is a mode of customer accommodation Pralahad and Ramaswamy (2008) liken unto *customization* – you can get a variant of the product, but one that is not truly personalized.

The Co-creation Approach

The Co-creation approach completely re-envisions customers as being a partner in the creation of value. Many of those with the highest reputation in the marketplace seem to have availed themselves of a personalizable relationship with their customers at the earliest possible moment via the Internet. Prahalad and Ramaswamy (2004c) and others (Kazman and Chen, 2009; Payne, et al., 2008) have each maintained that that co-creation goes beyond co-designing products and services; it establishes the mode under which the market will operate – a mode of equality. Such empowerment is evident in the open source software community, in crowd-sourcing, in Amazon's *Mechanical Turk* service, and in myriad other channel's for customer empowerment.

Thus, the attraction of a co-creative marketplace is in the value created, and in the values informing the interactions between firm and customer in a co-creative relationship. Figure 4 presents Prahalad and Ramaswamy's (2004c) building blocks to facilitate co-creative interactions between the firm and the customer. These principles read as though they are annotations from the Agile Manifesto (Fowler and Highsmith, 2001).

We can see the parallels between agility and co-creation, in terms of the principles each espouse, by focusing on Figure 4. Each of these "building blocks" are evident in both the manifesto and principles for agility, but also in the descriptions of many of agile methods (Boehm and Turner, 2004). Table 1 presents a comparison between the building blocks of developing a co-creative customer relationship and a generalization of similar principles from agile methods (particularly eXreme Programming). The similarities are striking in that realizations regarding changing markets, presented in 2000 by Prahalad and Ramaswamy from a marketing perspective, were also

described, in parallel, from a software development methods perspective (Fowler and Highsmith, 2001).

3. SERVICE-DOMINANT LOGIC

As the research and literature on agile methodologies and co-creation has progressed, other related concepts have evolved in the marketing literature. In particular, the concept of S-DL (Vargo and Lusch, 2004, 2008) refers to the shift in philosophy from a goods-dominant to service-dominant logic. This means that both goods and services (e.g. goods enhancement or the services offered by health, government, and education industries) should be viewed in terms of the services they provide.

From this perspective, there are two types of resources in a market or organization: *operant* and *operand*. Operand resources are those which must be acted upon in order to be beneficial. For example, operand resources in IT projects would refer to existing hardware and software which the organization owns and which can be useful in a new IT project. However, these operand resources are only useful if the project team has the appropriate operant resources—those which act on behalf of other resources to create value. In other words, the implicit knowledge held by systems development team members is necessary in order to produce the benefits of existing software components or hardware. Essentially, programmers use their operant resources (i.e. their knowledge and skills) to provide services to the IT project just as any provider does for its customers. We argue that additionally, in the co-creation model, a programmer also acts as an operand resource which is "acted upon" by the customer, who is an operant resource. The customer interacts with the programmer who is like an empty canvas waiting to be turned into personalized value. However, in the co-creation mode, the expertise of the programmer assists the customer by enabling a greater understanding of the palette of options available for the canvas.

There are ten foundational premises of S-DL (Vargo & Lusch 2008). We highlight the applicability of just three of those ten here (for brevity). First, service is the fundamental basis of exchange. In other words, the value produced by IT systems is viewed in terms of the service it provides to the customer—not the cost of the IT project or its on-time performance. Second (but fourth in Vargo and Lusch's [2008] list), operant resources are the fundamental source of competitive advantage. This is especially apparent in systems development. Any IT project team can get access to the hardware and software necessary to produce new IT systems. However, it is the ability of an IT project team to manipulate those operand resources which provides value to the customer. A project team's ability to actualize those resources to meet customer preference and needs is the primary source of competitive advantage over other project teams. Third (but sixth in Vargo and Lusch's [2008] list), in a service-oriented view of markets, the customer is a co-creator of value. In this case, where the customer also plays the role of the operant resource affecting an operand IT project team, the customer achieves competitive advantage in their own market by way of the degree to which they can use and manipulate an IT project team in order to produce a high-value IT system.

After understanding the foundational premises of S-DL, it is easy to view the shift toward agile and hybrid methodologies as a being part of a larger shift toward the service-oriented paradigm taking place everywhere, including the systems development market. While it would be an over-simplification to state that the sole reason for agile methodologies is to facilitate greater customer co-creation, agile methods are naturally well-suited to involve customers to a greater degree and to induce their input into the creative process more completely throughout the project life cycle.

Similarly, many organizations are not well-suited for agile methodologies, yet are searching for ways to adopt agile principles in their plan-driven techniques and are forming "hybrid" approaches. While there are a great many other factors (often related to project risk) which influence the methodology selection decision, we argue that this shift toward S-DL and co-creation is playing a large role whether directly or indirectly.

In summary, the S-DL view (Vargo & Lusch 2008) emerged in parallel to the co-creation concepts (Prahalad & Ramaswamy 2004a) in systems development, yet the two complement and inform each other. In the next section, we outline how these two conceptualizations affect extant theoretical models of methodology selection.

4. RE-CONCEPTUALIZED THEORETICAL MODEL OF METHODOLOGY SELECTION

Traditionally, methodology selection involves realizing a fit between the characteristics and assumptions of the systems development methodology and the degree of risk and uncertainty in the task environment (Barki et al. 2001). Once these risks were assuaged by the discipline of the systems development method, customer requirements would be met and customer value realized. However, Figure 6 inserts concepts of S-DL and Co-creation into the equation to suggest that customer involvement in design should also influence the fit that a method presents in a given problem domain and within a given set of task environment risks, uncertainties, and constraints. In other words, rather than switching methodologies, PMs may simply need to facilitate better co-creation into their process.

As we conceptualize how this new co-creative relationship will transpire between systems developers and customers, it is important to bear in mind that each party plays an equal-yet-distinct role in the partnership. This can be illustrated by Payne et al. (2008) in distinguishing between the role each party plays, and is also supported by other research into collaborative design between customers and software developers (Lee, 2007; Babb 2009). As it would be in the case of action research, participatory design, and other similar arrangements where dissimilar partners collaborate, the co-creative process is not necessarily one where both partners share the same expertise or concerns. According to Payne et al. (2008), the separation of concerns between the co-creative partners can be seen as a process ascribed to the customer, a process ascribed to the supplier, and the process of the encounters between them (Figure 7).

The encounter processes depicted in Figure 7 represent the important synergy made possible between the customer's processes and the supplier's processes during co-creation. Of particular interest is the inclusion of learning in the Payne et al. (2008) model. This model can be extended to incorporate the concerns, assumptions, and mechanics of systems development method selection in order to understand how this method facilitates the co-creation of value. Therefore, we must assume that these two parties, as they mutually construct meaning and value in their co-creation, each bring a unique perspective to the partnership. We can also conceive of these encounters between system developer and customer as an ongoing dialog, one in which value is created through exchanges of expertise and knowledge.

Re-conceptualizing Method Selection

We re-conceptualize the method selection process to include the co-creative concepts of S-DL and Co-creation. In Figure 8, we propose that systems development method selection involves choosing a technique which both fits the risk and uncertainty inherent in the task environment as well as fosters the necessary value co-creation with the customer—which is ultimately what determines project success, rather than time and cost.

Our model retains the importance of balancing the innate characteristics of a methodology with uncertainty and risk tolerance in the task environment. However, a co-creative partnership suggests that both the customer and systems developers stand to benefit in sharing the concerns of method selection, use, and evolution. In our conceptual model, the co-creation of value between customer and systems developer results in pathways of learning which allow the systems developer to realize an optimum methodological fit while the customer realizes increasing personalization. The value-creation encounters between customer and systems developer, facilitated by the chosen software development method, create new and unique operant resources for each party. For the systems developer, the operant resource is the new knowledge gained from their creative interactions with the customer. For the customer, the operant resource is the new knowledge concerning IT system capabilities and ideas for new systems potential. The co-created value to the developer is an optimized and tailored systems development method; and, the co-created value to the customer is the personalized IT system.

When we consider the Prahalad and Ramaswamy (2004a) building-blocks for establishing a co-creative relationship – dialog, access, risk-benefit, and transparency, it becomes easier to see why the demand for agile methods has arisen. Agile and hybrid methods are best-suited to facilitating a co-creative relationship between customer and systems developer. This is not to say that in all cases the customer-

provider relationship should be, or even can be, co-creative. However, in cases where co-creation is desired and/or warranted, agile methods appear to be the best fit thus far.

5. DISCUSSION AND IMPLICATIONS

We have proposed in this research that the evolution of agile and hybrid systems development methodologies are examples of a larger shift toward S-DL. Therefore, one of the primary implications is that project managers who are considering a switch to agile methodologies may need to pause and consider whether the cause of their current project failures are the result of traditional risk factors (e.g. lack of top management support or lack of operand resources such as knowledge/capabilities, etc.) or the result of poor customer co-creation when co-creation was in fact warranted. It is possible to involve customers to a greater degree using some form of hybrid methodology such as Boehm and Turner's *Incremental Commitment Model* (2004) or a service-oriented systems development technique (Keith et al., 2009) rather than forcing a more drastic change to a completely agile method. In addition, the S-DL perspective highlights the importance of operant resources as the primary source of competitive advantage over other systems developers. By choosing to outsource portions of a project, the PM is losing opportunities to develop new operant resources through customer co-creation.

Lastly, this research highlights the importance of valuing an IT project based on its ability to meet customer needs rather than create a product within a given time and cost constraint. Examples of large IT projects which are completed only to find that it doesn't meet the customer's needs are easy to find.

6. CONCLUSION

While this paper proposes that S-DL and Co-creation are reflections of changes in the task environment and market which demand new thinking in systems development methods, it is hard not to realize that agile methods are most in step with S-DL and Co-creation. If the same disruptive technology which prompted a revisit of our conceptualization of markets also prompted a revisit of systems development methods, then agile methods represents a swing in a pendulum in response to paradigmatic change. Eventually, as we are reminded by Kuhn (1996), stability in this new paradigm should arrive eventually. However, during the time of flux, as we have experienced for over a decade, practitioners and scholars of systems development methods would do well to understand changes in their own discipline – the advent of agile methods – through the experiences and wisdom of another discipline: marketing. Our re-conceptualization of systems development method selection accounts for the new and emerging relationship between customer and provider outlined in S-DL and Co-creation. We feel that this conceptualization opens up opportunities to study method selection and to study agile methods adoption and use in a new and useful light.

7. REFERENCES

Babb, J. (2009): Towards a Reflective-Agile Learning Model and Method In The Case Of Small-Shop Software Development: Evidence from an Action Research Study, PhD Dissertation, Virginia Commonwealth University, Richmond, VA.

Bardhan, I.R., Demirkan, H., Kannan, P.K., Kauffman, R.J., and Sougstad, R. (2010). An Interdisciplinary Perspective on IT Services Management and Service Science, *Journal of Management Information Systems*, 26, (4), 13-64.

Barki, H., Rivard, S., and Talbot, J. (2001). An integrative contingency model of software project risk management, *Journal of Management Information Systems*, 17, (4), 37-69.

Boehm, B. and Tuner, R. (2004). *Balancing Agility and Discipline: A Guide for the Perplexed*, Addison-Wesley, Boston. pp. 304.

Chow, T. and Cao, D. B. (2008). A survey study of critical success factors in agile software projects, *Journal of Systems and Software*, 81, (6), 961-971.

Dong, B., Evans, K.R., and Zou, S. The effects of customer participation in co-created service recovery, *Journal of the Academy of Marketing Science*, 36, 123-137.

Fowler, M. and Highsmith, J. (2001). The Agile Manifesto, *IEEE Software Development*, 9, (8), 28-35.

Giddens, A. (1984). *The Constitution of Society*, University of California Press, Berkeley, CA.

Kar, N.J. (2006). Adopting Agile Methodologies of Software Development: Agile Methodologies are Challenging Established Paradigms of Software Development, *Infosys SETLabs Briefings*, 4, (1), 3-10

Kazman, R. and Chen, H. (2009). The Metropolis Model: A New Logic for Development of Crowd-sourced Systems, *Communications of the ACM*, 52, (7), 76-84.

Keith, et al. (2009). Service-Oriented Software Development. *AMCIS 2009 Proceedings*. Paper 100.

Kuhn, T.S. (1996). *The structure of scientific revolutions (2nd Ed.)*, University of Chicago Press, Chicago.

Lee, A. S. (2007). Action is Artifact. In N. Kock, *Information Systems Action Research* (pp. 42-60). New York: Springer.

Madsen, S. & Matook, S. (2010). Conceptualizing Interpersonal Relationships in Agile IS Development, *Proceedings of the International Conference on Information Systems (ICIS2010)*, December 12-15, Saint Louis, MO, USA.

Payne, A.F., Storbacka, J., & Frow, P. (2008). Managing the co-creation of value, *Journal of the Academy of Marketing Science*, 36, 83-96.

Prahalad, C.K. and Ramaswamy, V. (2000). Co-opting Customer Competence, *Harvard Business Review*, 78, (1), 79-88.

Prahalad, C.K. and Ramaswamy, V. (2004a). Co-Creation Experiences: The Next Practice in Value Creation. Journal of Interactive Marketing, 18, 3, pp.14.

Prahalad, C.K. & Ramaswamy, V. (2004b). Co-creating unique value with customers, *Strategy & Leadership*, 32, (3), 4-9.

Prahalad, C.K. and Ramaswamy, V. (2004c). *The Future of Competition: Co-creating Unique Value with Customers*, Harvard Business School Publishing, Boston.

Ramaswamy, V. (2006). Co-Creating Experiences of Value with Customers, *Infosys SETLabs Briefings*, 4, (1), 25-36.

Sanders, E.B. & Stappers, P.J. (2008). Co-creation and the new landscapes of design, *CoDesign*, 4, (1), 5-18.

Saunders, C.S. & Scammel, R.W. (1986). Organizational power and the information services department: a reexamination. Communications of the ACM, 29, (2), 142-147.

Spohrer, J., Vargo, S.L., Caswell, N., and Maglio, P. (2008). The Service System is the Basic Abstraction of Service Science, *Proceedings of the 41st Hawaii International Conference on System Sciences*, January 7-10, Waikoloa, Big Island, Hawaii, USA.

Vargo, S.L. and Lusch R.F. (2004). Evolving to a New Dominant Logic for Marketing, *Journal of Marketing*, 68, 1-17.

Vargo, S.L. and Lusch R.F. (2008). Service-Dominant Logic: Continuing the Evolution, *Journal of the Academy of Marketing Science*, 36, 1-10.

Editor's Note:

This paper was selected for inclusion in the journal as the CONISAR 2011 Best Paper The acceptance rate is typically 2% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2011.

APPENDIX

Figure 1 Traditional Concept of a Market (Prahalad and Ramaswamy, 2004)

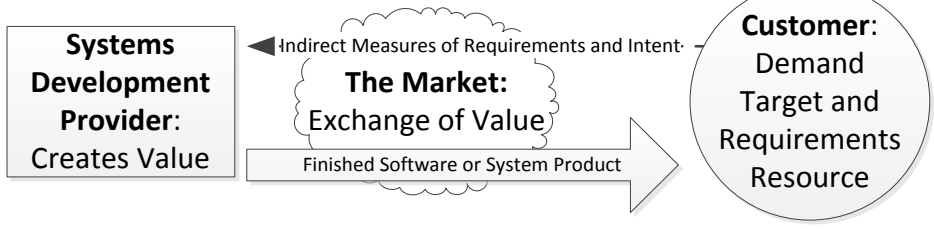


Figure 2 The Function of Traditional Software Development Methods

Figure 3 The Function of Traditional Software Development Methods (adapted from Pralahad and Ramaswamy, 2004c)

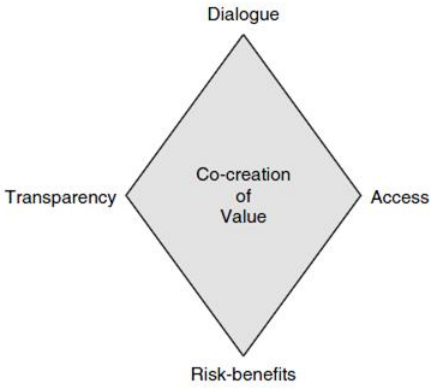


Figure 4 Building Blocks of Interactions for Co-creation of Value (Prahalad and Ramaswamy, 2004c)

	Co-creation	**Agile methods**
Dialogue	➢ Markets are a set of conversations between equal partners ➢ Joint Problem-Solvers ➢ Rules of Engagement for equality	➢ Design occurs between a client/developer partnership ➢ Regular interactions and releases of working software solves problems ➢ Regular meetings for equality
Access	➢ Firms provide reliable access in order to avoid information asymmetry	➢ Customer is a team member and is afforded regular access to discourage information asymmetry
Risk-benefits	➢ Firms provide reliable access in order to avoid information in order to foster a learning and empowering environment – Google: do no harm	➢ Both developer and customer are informed by participating in the process and each knows consequence and costs of change
Transparency	➢ Firm maintains an openness to customers in order to facilitate a dialog which creates value	➢ The customer is aware of release iterations and project velocity.

Table 1. A Sample Table

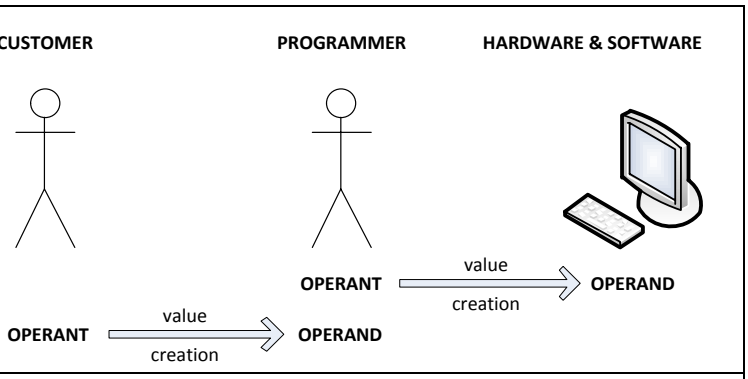


Figure 5 Types of Value Creation in Systems Development

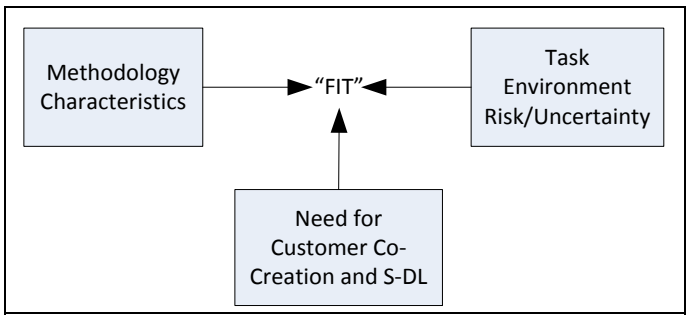


Figure 6 Nascent Theoretical Model of Methodology Selection Based on Co-Creation and S-DL

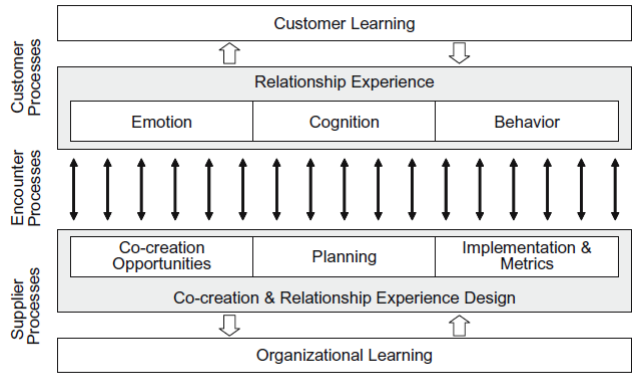


Figure 7 Conceptual Model for Value Co-creation (Payne et al., 2008)

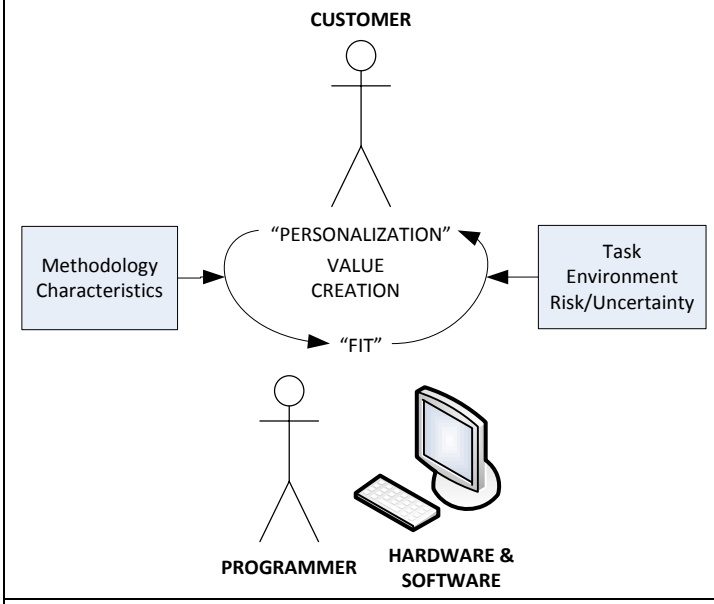


Figure 8 Re-conceptualized Theoretical Model of Methodology Selection

Open Source Software in the Vertical Market: An Open Niche?

Michael P. Conlon
michael.conlon@sru.edu
Computer Science Department,
Slippery Rock University of Pennsylvania
Slippery Rock, Pennsylvania 16057, U.S.A.

Abstract

Much of the universe of open-source software is categorized; abundant open-source software is found for most categories. However, relatively few dual-licensed open-source software programs are found, and very little open-source software is found for vertical markets. Explanations are explored.

Keywords: open source, vertical market, horizontal market, dual license

1. INTRODUCTION

The phrase *open source* has been in common use since it was suggested in 1998 by Christine Peterson as an alternative name for what many call *free software* (Open Source Initiative, 2007). This paper is an attempt to categorize each package of a large sample of open source software, so as to discover the domains in which open source development has been occurring, and in which domains, if any, there has been little or no open source development activity.

Much of the earliest open source software consisted of systems software: programming-language processors, utility programs, database management systems, and operating system kernels. For example, the author first downloaded a Linux distribution*, Soft Landing Systems (SLS) Linux* in 1992. (A Linux distribution consists of the Linux kernel, essential utility software such as programs to list, edit, rename, and delete files, other system software, and applications.) The SLS distribution contained a kernel (v. 0.99pl12), the command-line utilities, several language processors, the X-Window System, several programming libraries, but virtually no application software. It was clear at the time that, for Linux to become more-widely used, application software was needed.

Since then, much application software has been either written from scratch or has been open-sourced from previously-proprietary software. There has been substantial progress in developing more and better system software as well. So what potential domains for open-source software remain unexplored? That is the question this paper attempts to answer.

2. HYPOTHESES

The first hypothesis is that, in spite of the large variety of open-source software, very little of it would be vertical market software, i.e., software designed to automate businesses of a particular type. Thus, software for dentists' offices or software for plumbing businesses would be considered vertical-market software.

The second hypothesis is that most general business software would be dual-licensed. Several programs commonly used in business, such as *MySQL*, use the dual-licensing model so that the community of users of the open-source-licensed version can contribute improvements to the software (cutting development costs), and the company can sell support to licensees of the

proprietary-licensed version (providing a revenue stream).

3. DEFINITIONS

Both the Association for Computing Machinery, (1998) and the U.S. Patent and Trademark Office (2011) have developed classification schemes for software. For the purposes of this paper, however, popular classification terms were deemed more appropriate.

Several such categories of software are well-established, with the definition of the category generally agreed-upon. Some other categories are not as well-defined, perhaps because they were coined as marketing terms rather than as scientific categories. This paper will first define the category names so there will be no confusion.

Application software: software whose purpose is to solve users' problems. System software and application software are disjoint sets. Their union is the universe of software.

Art & Entertainment: software for creating, playing, or viewing graphic art, video, and/or music, or for entertaining the user. This category includes most game software, but this study did not examine game software.

Client: any software that requests services from servers. Clients are usually, but not always, interactive with users.

Cloud: any software that provides applications to users via the Worldwide Web. Such applications traditionally would have been provided locally on the user's computer.

Development software: software for creating, debugging, and/or maintaining software or websites.

Dual licensed: software distributed under an open-source license that is also available under a proprietary (non-open-source) license, typically for a fee.

General Business: software that typically would be used by businesses but not by individuals.

Graphics: software that is used to view, generate, or modify graphical art, photographs, or diagrams.

Horizontal market software: all software that is not vertical market software. Most horizontal market software would be of use to a variety of industries.

Music: software that is used to listen to, generate, modify, or notate music.

Operating System: An operating system kernel, or an operating system distribution (see below), provided the distribution is created by the entity that develops and maintains the kernel. This study does not include operating system distributions from third parties, since they are merely collections of software that may be examined separately.

Operating system distribution: a collection of software distributed as a unit, consisting of an operating system kernel, essential utility programs such as programs to list, edit, rename, and delete files, other system software, and applications.

PIM (Personal Information Manager): Email, calendar, collaborative communication, messaging, sticky note, and organizer software, etc., but not database managers.

Productivity: word processors, spreadsheet programs, presentation programs, small-office database management systems, and PDF viewers.

Server: any software that provides services to client software. Servers are never used by users directly; only client software may interact with a server.

System software: software whose purpose is to manage the computer, maintain the computer and its file system, or to help develop and debug software.

Utility: a program for maintenance or management of a computer system.

Vertical market software: software that is specialized to a particular industry, and that fully automates a company in that industry, or nearly so. There is much software that is specialized to just one aspect of a particular industry, and, in this paper, such software is not considered vertical market software.

Video: software that is used to view, generate, or modify moving images.

Web: any software that is involved, in any way, with the Worldwide Web. Such software could be client software, server software, or Web-development software.

4. METHODOLOGY

Selecting Software

There is so much open-source software that it is impractical to study it all. Therefore, one must rely on a sample. Eric Raymond (2000) stated, "The Linux world...has terabytes of open sources generally available." Freshmeat.net (2011) claims that "Thousands of applications, which are preferably released under an open source

license, are meticulously cataloged in the freshmeat database." And SourceForge (2011) claims to host 295,679 open-source projects.

While Freshmeat.net is the canonical listing of open-source software, it obtains its listings from the authors of the software, and so its listings are not vetted for utility, stability, practicality or popularity. Sourceforge serves as an archive for open-source projects, but a large fraction of its projects have had no activity for a substantial time (Rabellino, 2007), implying that they obtained no traction among open-source developers. Indeed, some have never reached version 1.0. Since this paper intends to study vibrant projects, the sample of software must be defined by individuals or organizations independent of the software's creators.

The author was able to find three independent lists of open-source software. Wikipedia (2011) and Harvey (2011) each listed a significant number of open-source packages, and all of them were included in this study. The *Google Summer of Code* (GSOC) (Google, 2011) has supported a large number of projects. All projects involved with GSOC 2005 and most from GSOC 2006 were studied.

For each selected open-source project, the author inspected the project Website and the Website of the referring site. Each site was analyzed to determine into which categories (from section *3* above) the project's software belonged. Not every Website supplied explicitly the information needed for this study. In the small number of cases where the Website was vague, the value for the category was inferred from contextual information in the Websites.

The Spreadsheet

Each open-source package is represented by a row in the spreadsheet. A column was created for each of many software categories, although *vertical market, horizontal market*, and *dual licensed* were of primary interest. If the package seemed to fit the category, a "Y" was entered into the cell at the junction of the package's row and the category's column.

5. RESULTS

As indicated in the table in the appendix, only 5% of the software packages in the sample of one hundred eighty-four were vertical-market software, confirming the hypothesis that open-source vertical-market software would be rare. 5% of the packages in the sample is actually large compared with the percent of industries represented. The 2007 North American Industry Classification System (United States Census Bureau, 2011) lists 1,175 industry categories. Our sample identifies only five industries with open-source, vertical-market (OSVM) software: library, microfinance, tool-and-die, restaurant, and financial services. This computes to 0.43% of all industries.

Only eleven of the forty-nine (22%) of general business software were dual-licensed. Even if general business software where commercial hosting is available from the vendor is counted as dual-licensed, the figure is still only 33%, and the hypothesis that most general business software would be dual licensed is not supported by the data in this sample.

6. DISCUSSION

Vertical Market Software

What explains the scarcity of vertical market open-source software? Eric Raymond (2000) postulated that "Every good work of software starts by scratching a developer's personal itch." When the developer is a hobbyist, he is not likely to be itched by the desire to write dentist-office software, and the chances are that he wouldn't know where to start, unless he were a dentist himself. If he is a dentist, and the software development project was successful, significant money might be earned by licensing the software to other dentists, an incentive to make the software proprietary. If he did make it open-source, he would be offering competitors the ability to operate as efficiently as he does, for no development or licensing cost: not a wise decision in a competitive industry. For a detailed discussion of the obstacles facing open-source projects in vertical markets, refer to Shaffer (2006).

Thus, the domain knowledge combined with the software design talent required to create good vertical market software must be a relatively rare combination, and those that have such knowledge have significant disincentives against open-sourcing their creation.

Nonetheless, this study did find several open-source, vertical-market packages. What factors led to their creation in the face of the above-mentioned disincentives?

Five of the ten were integrated library systems. (ILS's). Two others were microfinance software. Of the remainder, *Floreant POS* is point-of-sale software for restaurants, *Tool and Die ERP* is for tool-and-die companies, and *OpenGamma* is for financial analytics at investment companies.

The existence of the library information systems is easy to explain. As the former treasurer of a small-town, one-room public library, the author was greatly disturbed by the $2000 annual ILS license fee, particularly since this was one-sixth of the library's annual budget. Vertical-market software is notoriously high-priced, and these prices create a significant incentive for a library to find a more economical source for ILS software.

The principles of library operation are more generally understood than those of less-public ventures, so there should be more people competent to create an ILS than, for example, an integrated dentist-office system. As non-profit organizations or government entities, libraries would not find it appropriate to initiate a profit-making software business. Additionally, and unlike for-profit firms, libraries do not generally compete with one another; therefore, a library that creates its own ILS would not be at any disadvantage should other libraries adopt their software. Under an open-source regime, the library that initiates the ILS software project may find their software enhanced by other libraries, with all user-libraries reaping the benefits. Thus many obstacles to the creation of open-source vertical-market software do not exist in the library domain.

The *Koha* ILS illustrates this. Horowhenua Library Trust (HLT), which manages several public libraries in New Zealand, faced the Y2K problem on their existing ILS. They distributed an RFP for a replacement system, but found nothing adequate and affordable among the submitted bids. Thereupon, they decided to create a new open-source ILS from scratch, and hired Katipo Communications, a Web software development firm, to help them create it. The new software became operational in just over fifteen weeks, through intense cooperation between Katipo and HLT's librarians. They called it *Koha*, and they created it for 40% of the cost of the average turnkey solution (Ransom, Cormack, and Blake, 2009).

Other libraries worldwide have contributed improvements to Koha, and all the libraries that use it can take advantage of the enhanced software. HLT, at relatively low initial cost, has broken free of the lock-in and concomitant high licensing fees of proprietary ILS's, and has acquired a high-quality, free (from onerous licensing conditions), open-source ILS (Ransom et. al., 2009).

In addition to the five ILS's, two microfinance programs were found: *Mifos* and *Octopus*. Mifos was developed by the Grameen Foundation, and Octopus by the Agency for Technical Cooperation and Development (ACTED). Both organizations are charitable organizations rather than profit-making businesses, and their goal is to promote microfinance.

Tool and Die ERP is enterprise resource management software for the tool and die industry. It was created under the sponsorship of the European Union to help improve the competitiveness of European tool-and-die firms. As a government project, the *Tool and Die ERP* project had no concerns about inadvertently sharing competitive advantage with other firms.

Each of the projects discussed thus far seems to owe its success to its immunity to the disincentives that generally stifle open-source vertical-market (OSVM) software. Are there any other circumstances under which OSVM software can arise?

FloreantPOS is a point-of-sale system for restaurants. It was developed by Moonrank U.S.A., a Web software development firm. Their Website does not reveal the motivation for FloreantPOS's development, but it does seem that Moonrank expects to profit by providing support (Moonrank, 2011). FloreantPOS claims at least one major restaurant chain, *Denny's*, as a client. It is not clear how FloreantPOS has overcome the disincentives against OSVM. Perhaps restaurants, or at least those restaurants that are FloreantPOS users, consider their food and ambiance greater differentiators than their IT systems. Attempts to contact Moonrank for further information were unsuccessful.

The last OSVM to be discussed is an interesting new project that has been initiated by OpenGamma, a startup company. OpenGamma is developing software for the front office and risk analysis functions of Wall Street firms. They

believe that these functions have become sufficiently standardized that they no longer provide significant competitive advantage to Wall-Street firms, and that it will be cheaper for companies to use OpenGamma's open-source program and pay for support than to license third-party software or develop and maintain their own (Woods, 2011). They will depend on dual-licensing and confidentiality agreements to assure their clients that their trade secrets will not be compromised.

As of the date of writing, September 2011, OpenGamma has not reached version 1.0, (OpenGamma, 2011a), and until that point is reached, one would not expect it to be used as a production system. The OpenGamma Website indicates that they are "trialing it with a number of financial institutions" (OpenGamma 2011b), but that is no guarantee that any significant institutions will become production users. While venture-capitalists are betting on OpenGamma, the low success rate of VC-funded firms, about 45% (Davis, 2008) precludes any assumption that VC funding necessarily predicts success.

Dual-licensed Software

There appears to be a relative scarcity of dual-licensed open-source software. MySQL, SugarCRM, Zimbra, and Bacula are high-profile dual-licensed open-source projects. As profit-making endeavors, these projects need to stimulate public interest through advertising and press releases. Hearing about such projects regularly may leave the impression that dual-licensed projects are more common than they actually are.

Dual-licensing software is a proposed solution to the problem of making profits from free, open-source software. The rapid rise of MySQL showed that such a business model could both generate profits and produce rapidly-improving software. MySQL's success also gave the model significant exposure, leading other enterprises to imitate. However, this survey suggests that there are not very many companies replicating MySQL's success.

7. CONCLUSIONS

Open-source software has limited penetration into the vertical-market world. Most of the existing OSVM software rely on government sponsorship or their situation in a noncompetitive industry for their success. However, there are OSVM applications for competitive markets, and at least one of these has met with significant acceptance. It seems likely that open-source software will move further into vertical markets only if exceptions are found to the disincentives to OSVM software.

The exceptions so far identified include

a) government or non-profit sponsorship,
b) territorially-segregated or other non-competitive markets,
c) ability to profit from selling support,
d) potential cost savings from avoiding license fees of proprietary software and sharing the development burden with your industry, and, perhaps,
e) maturing technology eliminating the competitive advantage of proprietary software technology.

In those cases where several of these factors are present, the emergence of open-source software in that vertical market would be more likely.

Firm conclusions about dual-licensed software are harder to come by. It could be that MySQL's success owed much to timing: arising just as the World-Wide Web and e-commerce were emerging, when an alternative to expensive and bloated commercial databases was particularly needed, MySQL met a need.

It may be that users of open-source software distrust mixed-model (another term for dual-licensing) companies, but that the need was so great at MySQL's emergence that the part-proprietary aspect was overlooked. It is certainly possible that other software might find a similar niche at emergence, and so there are other successful dual-licensed projects. However, if this conjecture is true, a mixed-model project should find it hard to compete against a pure, community-developed open-source project. It would be intriguing to study the several dual-license projects, and their competitive environment, to elucidate which ones are truly successful and why.

8. REFERENCES

Association for Computing Machinery (1998). The 1998 ACM Computing Classification Scheme. Retrieved June 7, 2011 from http://www.acm.org/about/class/ccs98-html.

Davis, M.P. (2008). VC Backed Startup Success Rate. Retrieved June 14, 2011 from

www.markpeterdavis.com/getventure/2008/09/vc-backed-start.html.

Freshmeat.net (2011). About freshmeat.net. Retrieved June 11, 2011 from http://freshmeat.net/about.

Google (2011). Google Summer of Code. Retrieved June 11, 2011 from http://code.google.com/soc.

Harvey, C. (2011) 70 Open Source Replacements for Small Business Software. Datamation, April 19, 2011. Retrieved June 7, 2011 from http://itmanagement.earthweb.com/osrc/article.php/12068_3931181_1/70-Open-Source-Replacements-for-Small-Business-Software.htm.

Karaguchi, S., Garg, K., Matsushita, M., & Inoue, K. (2004). MUDABlue: an automatic categorization system for open source repositories. ***APSEC '04:*** *Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC 2004)*, 184-193.

Moonrank, U.S.A., L.L.C. (undated). FloreantPOS Home Page. Retrieved June 14, 2011 from http://floreantpos.com.

Open Source Initiative (ca. 2008). History of the OSI. Retrieved June 3, 2011 from http://opensource.org/history

OpenGamma (2011b). *Blog* page. Retrieved September 5, 2011 from http://www.opengamma.com/blog

OpenGamma (2011a). *Developers* page. Retrieved June 10, 2011 from http://developers.opengamma.com

Rabellino, G (2007). Lies, Damn Lies, and Sourceforge Statistics. *Boldly Open*, April 4, 2007. Retrieved June 14, 2011 from http://boldlyopen.com/2007/04/04/lies-damn-lies-and-sourceforge-statistics/

Raymond, Eric S. (2000). The Cathedral and the Bazaar. Retrieved June 7, 2011 from http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/index.html

Ransom, J., Cormack, C., and Blake, R. (2009). How Hard Can It Be? : Developing in Open Source. Code{4}lib Journal, (7), June 26, 2009. Retrieved on June 12, 2011 from http://journal.conde4lib.org/articles/1638.

Shaffer, George (2006). The Limits of Open Source - Vertical Markets Present Special Obstacles. *GeodSoft Website*. Retrieved June 10, 2011 from http://geodsoft.com/opinion/oslimits/vertical.htm.

Sourceforge (2011). Sourceforge.net homepage. Retrieved June 11, 2011 from http://sourceforge.net.

United States Census Bureau (2011). The North American Industry Classification System (2007 NAICS). Retrieved Sept. 5, 2011 from http://www.census.gov/cgi-bin/sssd/naics/naicsrch?chart=2007.

United States Patent and Trademark Office (2011). The U.S. Patent Classification System. Retrieved June 7, 2011 from http://www.uspto.gov/web/offices/documents/classescombined.pdf.

Wikipedia (2011). List of Free and open source software packages. Retrieved June 7, 2011 from http://en.wikipedia.org/wiki/List_of_free_and_open_source_software_packages.

Woods, Dan (2011). Open Source for Vertical Apps: Is Wall Street Ready? Forbes' *CIO* Central, June 8, 2011. Retrieved June 10, 2011 from http://blogs.forbes.com/ciocentral/2011/06/08/open-source-for-vertical-apps-is-wall-street-ready.

Appendix

Software	Type																		Web Resource	Description
Name	Vertical Market	Horizontal Market	Art&Entertainment	System	Operating System	Utility	Development	Server	Application	Client	Web	Cloud	Productivity	PIM	General Business	Dual License	Middleware	Graphics	Video	Music		
7-Zip		Y		Y		Y															www.7-zip.org	Compress/archive utility
AbiWord		Y											Y								www.abisource.com	Word processor
Adempiere		Y						Y	Y		Y	Y			Y						www.adempiere.com	ERP software
Adium		Y								Y											adium.im	Instant messaging client
Alfresco		Y						Y	Y		Y	Y			Y	Y					www.alfresco.com	Content management system
Amanda		Y		Y		Y									Y						amanda.zmanda.com	Backup utility
Apache HTTP Server		Y		Y				Y	Y		Y	Y									httpd.apache.org	Web server
Ardour		Y	Y																	Y	ardour.org	Audio/music processor
Areca Backup		Y		Y		Y															www.areca-backup.org	Backup utility
Argentum		Y						Y	Y		Y	Y			Y	1		Y			www.argentuminvoice.com	Invoicing software
ArgoUML		Y		Y			Y											Y			argouml.tigris.org	Uniform Modeling Language s/w
Ascalaph Designer		Y							Y									Y			www.biomolecular-modeling.com/Ascalaph	Molecular modeling and dynamics
Asterisk		Y						Y	Y						Y						www.asterisk.org	Digital PBX (VOIP)
Audacity		Y	Y						Y											Y	audacity.sourceforge.net	Audio/music processor
Avogadro		Y							Y									Y			avogadro.openmolecules.net	Molecular modeling
Bacula		Y		Y		Y										Y					www.baculasystems.com	Backup utility
Bioclipse		Y								Y											www.bioclipse.net	Bio/Chem Informatics
BioRails		Y							Y									Y			biorails.org	Bioinformatics
Blender		Y	Y						Y									Y			www.blender.org	3D graphics software
Boo		Y		Y				Y													boo.codehaus.org	Programming language compiler
Bookyt		Y						Y	Y		Y	Y			Y						redmine.cyt.ch/projects/bookyt	Double-entry bookkeeping
Boost C++		Y		Y				Y			Y	Y									www.boost.org	Programming libraries for C++
Bricolage		Y						Y	Y		Y	Y									bricolagecms.org	Content management system
Broadleaf Commerce		Y							Y		Y				Y						www.broadleafcommerce.org	eCommerce software
CellProfiler		Y												Y				Y			cellprofiler.org	Microscope image processing s/w
Chemistry Development Kit		Y		Y				Y						Y							cdk.sourceforge.net	Cheminformatics library
Chrome		Y							Y	Y	Y										www.google.com/chrome	Web browser
Cinelerra		Y	Y						Y										Y		www.cinelerra.org	Video editing software
ClamAV		Y		Y		Y															www.clamav.net	Anti-virus filter for servers
ClearOS		Y		Y	Y			Y								Y					www.clearfoundation.com/Software/overview.html	General-purpose network server
Collabtive		Y						Y	Y		Y	Y		Y	Y	1					collabtive.o-dyn.de	Groupware
CUPS		Y		Y		Y		Y	Y												www.cups.org	Unix printing system
Compiere ERP and CRM		Y													Y	Y					www.compiere.com	ERP & CRM software
DaisyCMS		Y		Y		Y		Y	Y		Y	Y									www.daisycms.org	Content management system
Dia		Y							Y									Y			live.gnome.org/Dia	Diagramming tool
Django		Y		Y		Y		Y	Y		Y										djangoproject.com	Web framework (CMS?)
Dojo		Y		Y		Y		Y	Y		Y										dojotoolkit.org	Javascript toolkit
Drupal		Y		Y		Y		Y	Y		Y	Y									drupal.org	Content management system
Eclipse		Y		Y		Y		Y	Y		Y	Y									www.eclipse.org	Integrated development environment
EdgeERP		Y							Y		Y	Y			Y						www.edgeerp.org	ERP software
Edoceo Imperium		Y										?			Y						imperium.edoceo.com	Accounting and business mgmt. s/w
eHour		Y							Y		Y	Y			Y						www.ehour.nl	Timesheet management
Endian Firewall		Y		Y		Y			Y							Y					www.endian.com	Firewall
Endrov		Y		Y				Y	Y		Y	Y						Y			www.endrov.net	Microscope image proc'ing apps & lib
ERP5		Y									Y	Y			Y	1					www.erp5.com	ERP software
Evergreen	Y								Y		Y	Y									evergreen-ils.org	Integrated Library System
Evolution		Y							Y		Y			Y							projects.gnome.org/evolution	Personal information manager
FFmpeg		Y		Y				Y											Y	Y	ffmpeg.org	audio/video programming library
FIJI		Y							Y									Y			pacific.mpi-cbg.de/wiki/index.php/Fiji	Microscope image processing s/w

Software	Type																					Web Resource	Description
Name	Vertical Market	Horizontal Market	Art&Entertainment	System	Operating System	Utility	Development	Server	Application	Client	Web	Cloud	Productivity	PIM	General Business	Dual License	Middleware	Graphics	Video	Music			
Firefox		Y							Y	Y	Y											www.mozilla.org	Web browser
Floreant POS	Y							Y	Y	Y												floreantpos.com	Point-of-sale software for restaurants
FreeBSD		Y		Y	Y																	www.freebsd.org	Operating system
FreeMind		Y							Y													freemind.sourceforge.net	Mind-mapping software
Freenet		Y						Y	Y		Y											freenetproject.org	Anonymous information server
Frescobaldi		Y	Y						Y											Y		www.frescobaldi.org	Music notation editor
Front Accounting		Y						Y	Y		Y	Y			Y			Y				frontaccounting.com	ERP for small companies
Gallery		Y	Y					Y	Y		Y	Y										gallery.menalto.com	Photo album organizer
GanttProject		Y							Y						Y			Y				www.ganttproject.biz	Project-management software
GCC		Y		Y			Y															gcc.gnu.org	Programming language compilers
GENtle		Y							Y									Y				gentle.magnusmanske.de	Bioinformatics
Get Simple		Y							Y		Y				Y							get-simple.info	Content management system
GhostScript/GhostView		Y							Y				Y					Y				pages.cs.wisc.edu/~ghost	Postscript/PDF viewer/printer
GIMP		Y	Y						Y									Y				www.gimp.org	vector graphics processor
Gnome		Y		Y					Y													www.gnome.org	GUI desktop environment
Gnu Cash		Y							Y						Y							www.gnucash.org	Personal finance manager
Gnu Utilities		Y		Y		Y																directory.fsf.org/GNU	Unix shell commands
Gnumeric		Y							Y				Y									projects.gnome.org/gnumeric	Spreadsheet
gPXE		Y		Y		Y																etherboot.org	Network bootloader
Grass		Y						Y	Y	Y								Y				grass.fbk.eu	Geographic information system
Grisbi		Y							Y													www.grisbi.org	Personal finance manager
GROMACS		Y							Y									Y				www.gromacs.org	Molecular dynamics
Group-Office		Y		Y				Y	Y		Y	Y		Y	Y							www.group-office.com	Groupware
Haskell		Y		Y			Y															haskell.org	functional programming language
HomeBank		Y							Y													homebank.free.fr	Personal finance manager
Horde		Y		Y			Y	Y	Y	Y	Y	Y		Y								www.horde.org	Web/cloud PIM framework and apps
ImageJ		Y	Y						Y									Y				rsb.info.nih.gov/ij	Microscope image processing s/w
InkScape		Y	Y						Y									Y				inkscape.org	SVG editor
Int'l Components for Unicode		Y		Y			Y															site.icu-project.org	Unicode programming library
Jabber		Y						Y	Y	Y	Y											codex.xiaoka.com/wiki/jabberd2:start	Instant messaging server
jBoss		Y							Y								Y					www.jboss.org	Middleware
jFin		Y		Y			Y								Y							jfin.org	Derivatives trade processing library
Jfire		Y						Y	Y	Y	Y				Y							www.jfire.net	ERP, CRM, & Framework
Jgnash		Y							Y													sourceforge.net/projects/jgnash	Personal finance manager
JquantLib		Y		Y			Y								Y			Y				www.jquantlib.org	Quantitative finance library
Jmol		Y		Y			Y		Y		Y							Y				www.jmol.org	Molecular modeling
JOELib		Y							Y													sourceforge.net/projects/joelib	Cheminformatics
Joomla		Y		Y				Y	Y	Y	Y	Y										opensourcematters.org/joomla.html	Content management system
K-meleon		Y							Y	Y	Y											kmeleon.sourceforge.net	Web browser
Kamaelia		Y		Y			Y															www.kamaelia.org	Concurrency library
KDE		Y		Y					Y										Y			www.kde.org	GUI desktop environment
Kdenlive		Y	Y						Y										Y			www.kdenlive.org	Video editing software
Kino		Y	Y						Y										Y			www.kinodv.org	Video editing software
KmyMoney		Y							Y						Y							kmymoney2.sourceforge.net	Personal finance manager
Koffice		Y							Y				Y									www.koffice.org	Office suite
Koha	Y							Y	Y		Y	Y										koha-community.org	Integrated Library System
KompoZer		Y		Y			Y				Y											kompozer.net	Web page editor
LAMMPS		Y							Y									Y				lammps.sandia.gov	Molecular dynamics
Lazy8 Ledger		Y							Y			?			Y							lazy8.nu/lazy8ledger	Bookkeeping software

Software	Type																				Web Resource	Description
Name	Vertical Market	Horizontal Market	Art&Entertainment	System	Operating System	Utility	Development	Server	Application	Client	Web	Cloud	Productivity	PIM	General Business	Dual License	Middleware	Graphics	Video	Music		
LedgerSMB		Y							Y			?			Y						www.ledgersmb.org	Small business accounting software
Lemon POS		Y						Y	Y	Y					Y						www.lemonpos.org	Point-of-sale software
Libre/Open Office		Y							Y				Y								www.libreoffice.org	Office suite
LilyPond		Y	Y						Y									Y		Y	lilypond.org	Music notation processor
Linux		Y		Y	Y																www.kernel.org	Operating system kernel
LiVES		Y	Y						Y										Y		lives.sourceforge.net	Video editing software
MDynaMix		Y							Y									Y			people.su.se/~lyuba/mdynamix	Molecular dynamics
MeshLab		Y							Y									Y			meshlab.sourceforge.net	Mesh processing (3-D graphics) s/w
Mifos	Y							Y	Y		Y	Y		1							mifos.org	Microfinance software
MindTouch		Y						Y	Y		Y	Y			Y	Y					www.mindtouch.com	Business intelligence software
Molekel			Y						Y									Y			molekel.cscs.ch	Molecular modeling
Mono		Y		Y			Y	Y	Y												www.mono-project.com	Byte-code interpreter and compilers
Monotone		Y		Y		Y	Y	Y													www.monotone.ca	Distributed version control system
Mplayer		Y	Y						Y										Y		www.mplayerhq.hu	Digital audio player
MuseScore		Y	Y						Y									Y		Y	musescore.org	Music notation processor
MySQL		Y		Y				Y	Y							Y					mysql.com	Database management system
NAMD		Y							Y									Y			ks.uiuc.edu/Research/namd	Molecular dynamics
NetBSD		Y		Y	Y																www.netbsd.org	Operating system
NewGenLib	Y							Y	Y		Y	Y				Y					www.verussolutions.biz	Integrated Library System
Nmap		Y		Y		Y					Y										nmap.org	Network security scanner
nopCommerce		Y							Y						Y						www.nopcommerce.com	E-commerce software
NSIS		Y		Y		Y															nsis.sourceforge.net	Windows software installer generator
NVU		Y		Y			Y				Y										net2.com/nvu	Web design software
Octopus Microfinance Suite	Y									Y											www.octopusnetwork.org	Microfinance software
OFBiz		Y						Y	Y	Y		Y	Y		Y						ofbiz.apache.org	ERP software
OpenBabel		Y		Y			Y		Y												openbabel.sourceforge.net	Cheminformatics
OpenBiblio	Y							Y	Y	Y	Y	Y	Y								obiblio.sourceforge.net	Integrated Library System
Openbravo ERP		Y						Y	Y	Y	Y	Y	Y		Y	Y					forge.openbravo.com	ERP software
Openbravo POS		Y						Y	Y	Y	Y	Y	Y		Y						forge.openbravo.com	Point-of-sale software
OpenBSD		Y		Y	Y																www.openbsd.org	Operating system
OpenERP		Y						Y	Y	Y	Y	Y	Y		Y	1					www.openerp.com	ERP software
OpenGamma	Y							Y								Y					www.opengamma.com	Financial analytics calc and delivery
OpenProj		Y							Y		Y	Y	Y		Y						openproj.org/openproj	Project management software
opentaps		Y						Y	Y		Y	Y	Y		Y						www.opentaps.org	ERP & CRM software
Orange HRM		Y						Y	Y		Y	Y	Y		Y	1					www.orangehrm.com	HRM software
OSCAR		Y		Y		Y															svn.oscar.openclustergroup.org	Sets up a Beowulf computer cluster
osCommerce		Y						Y	Y		Y				Y						www.oscommerce.com	Retail e-commerce software
PeaZip		Y		Y		Y															www.peazip.org	File and archive manager utility
Perl		Y		Y			Y														www.perl.org	Programming language interpreter
phpGroupWare		Y						Y	Y		Y	Y	Y		Y						savannah.gnu.org/projects/phpgroupware	Groupware
Phreedom		Y						Y			Y	Y			Y						www.phreesoft.com	ERP software
Pidgin		Y							Y	Y											www.pidgin.im	Instant messaging client
PMB	Y								Y	Y	Y	Y									www.pmbservices.fr	Integrated Library System
PrestaShop		Y						Y	Y		Y	Y			Y						www.prestashop.com	Retail e-commerce software
PyMol		Y							Y									Y			pymol.org	Molecular modeling
Python		Y		Y			Y														www.python.org	Programming language compiler
QCAD		Y							Y							Y		Y			www.qcad.org	CADD
QuickFIX/J		Y		Y			Y														www.quickfixj.org	Financial Info eXchange protocol lib
QuteMol		Y							Y									Y			qutemol.sourceforge.net	Molecular modeling

Software Name	Type: Vertical Market	Horizontal Market	Arts&Entertainment	System	Operating System	Utility	Development	Server	Application	Client	Web	Cloud	Productivity	PIM	General Business	Dual License	Middleware	Graphics	Video	Music	Web Resource	Description
RasMol		Y							Y							Y		Y			www.rasmol.org	Molecular modeling
Rosegarden		Y	Y						Y									Y		Y	www.rosegardenmusic.com	Music processor
Ruby		Y		Y			Y														www.ruby-lang.org	Object-oriented scripting language
Scribus		Y							Y									Y			www.scribus.net	Desktop publishing software
ShoutCast		Y	Y					Y	Y											Y	www.shoutcast.com/broadcast-tools	Internet "radio" station software
Simple Invoices		Y						Y	Y	Y	Y	Y			Y						www.simpleinvoices.org	Invoicing software
SimPy		Y		Y		Y	Y														simpy.sourceforge.net	Discrete event simulation package
Siwapp		Y						Y	Y		Y	Y									www.siwapp.org	Invoicing software
Sonar		Y		Y			Y	Y	Y	Y	Y	Y	Y								www.sonarsource.org	Code quality management software
SplendidCRM		Y						Y	Y	Y	Y	Y			Y	Y					www.splendidcrm.com	CRM software
SQL-Ledger		Y											Y	Y							www.sql-ledger.com	Ledger and ERP software
Subversion		Y		Y		Y	Y	Y							Y	Y					subversion.apache.org	Distributed version control system
SugarCRM		Y						Y	Y		Y	Y			Y	Y					www.sugarcrm.com	CRM software
The GIMP		Y	Y						Y									Y			www.gimp.org	Raster graphics editor
Thunderbird		Y							Y	Y			Y	Y							www.mozillamessaging.com/en-US/thunderbird	Personal information manager
TimeTrex		Y						Y	Y		Y	Y			Y	Y					www.timetrex.com	Worker time and attendance tracker
Tinker		Y							Y									Y			dasher.wustl.edu/tinker	Molecular dynamics
Tk/Tkl		Y		Y			Y														tcl.sourceforge.net	Programming language interpreter
Tool and Die ERP	Y							Y	Y												toolanddie.sourceforge.net	ERP for tool-and-die companies
Tryton		Y						Y	Y	Y					Y						www.tryton.org	ERP software
TurboCash		Y							Y						Y	Y					turbocash.net	SMB finances software
TuxType		Y							Y												tux4kids.alioth.debian.org/tuxtype/	Typing tutor
UGENE		Y							Y			Y						Y			ugene.unipro.ru	Bioinformatics
Umbrello		Y		Y			Y											Y			www.umbrello.org	CASE software
Untangle		Y		Y		Y		Y								Y					www.untangle.com	Network mgmt. server
VoltDB		Y		Y			Y	Y								Y					voltdb.com	Database management system
vTiger		Y							Y	Y	Y	Y			Y						www.vtiger.com	CRM software
webERP		Y							Y	Y	Y	Y			Y						www.weberp.org	ERP software
WinAmp		Y							Y							Y				Y	www.winamp.com	Digital audio player
Wine		Y		Y				Y	Y								Y				www.winehq.og	Windows compatibilty layer
World Wind		Y						Y	Y									Y			www.freeearthfoundation.com	Virtual globe server (GIS)
xTuple PostBooks		Y						Y	Y	Y					Y	Y					www.xtuple.com	ERP software
Xwiki		Y						Y	Y		Y										www.xwiki.org	Wiki development platform & apps
Zen Cart		Y						Y	Y		Y	Y			Y						www.zen-cart.com	E-commerce shopping-cart software
Zentyal		Y		Y	Y			Y			Y	Y									www.zentyal.org	SMB multipurpose server
Zimbra		Y						Y	Y	Y	Y	Y	Y	Y	Y	Y					www.zimbra.com	SMB multipurpose server
Zinf		Y	Y						Y											Y	www.zinf.org	Digital audio player
App count	184	1. Not dual licensed, but commercial hosting is available from the vendor.																			100%	
	Count	10	174	18	60	6	21	37	71	130	21	61	49	6	7	49	30	2	39	6	10	
	Fraction	.05	.95	.10	.33	.03	.11	.20	.39	.71	.11	.33	.27	.03	.04	.27	.16	.01	.21	.03	.05	

Measuring Propagation in Online Social Networks: The Case of YouTube

Amir Afrasiabi Rad
a.afrasiabi@uOttawa.ca
School of Electrical Engineering and Computer Science

Morad Benyoucef
Benyoucef@Telfer.uOttawa.ca
Telfer School of Management

University of Ottawa
Ottawa, Ontario K1N 6N5, Canada

Abstract

We conducted a propagation analysis on an open social network, i.e., YouTube, by crawling one of its friendship networks and one of its subscribers (followers) networks. Our study is unique because it investigates the two main types of connections (i.e., friends and followers) within the same environment and interaction features. We observed that the effect on propagation of people who are not either in a friendship network or a subscription network is higher than that of friends or subscribers. Meanwhile, we found that even though the network of subscribers was denser than the network of friends, the magnitude of propagation in the subscription network was less than in the friendship network. We also noticed a low correlation between the popularity of content and its propagation in general, with a greater correlation in subscription networks than that in friendship networks.

Keywords: Social Network, Social Link Type, Information Propagation, Viral Marketing, YouTube

1. INTRODUCTION

Social networking websites, such as MySpace, Facebook, Twitter, Flickr, Orkut, YouTube, etc. are becoming more and more popular. To illustrate this popularity, it is enough to refer to social networks' usage statistics. In the US alone, social networks attracted more than 90% of all teenagers and young adults (Trusov, Bodapati, & Bucklin, 2010). More than 35 hours of videos are uploaded on YouTube every minute (YouTube LLC., 2010); and over 750 million active Facebook users share more than 30 billion pieces of content, and spend over 23 billion minutes on Facebook every month (Facebook Inc., 2011). The increase in the user population of social networks leads to a rise in user interaction, and ultimately higher volumes of generated and distributed content. This massive popularity of social networks, and their high user-base and user participation rates, along with the enormity and variety of user generated content turned social networking sites into hubs of social activity, and shaped them into a new generation of information mediums. Moreover, the interconnectivity of users in online social networks allows user generated content to be easily propagated through the whole social network.

The above mentioned facts attracted the attention of the marketing community. These unique characteristics of social networks provide the opportunity to harness the collective opinions of the population in order to shape user behavior and design marketing campaigns while

gaining insights about future market trends (Asur & Huberman, 2010; Bearden, Calcich, Netemeyer, & Teel, 1986; Leskovec, Adamic, & Huberman, 2007). Furthermore, the possibility of content propagation along the social links builds a huge community of users who can be seen as viral advertisers. Many studies have been conducted to analyze the opportunities of viral advertisement on social networks (Bearden et al., 1986; Van den Bulte & Joshi, 2007; Domingos & Richardson, 2001; Duan, Gu, & Whinston, 2008; Evans, 2009; Hu, Tian, Liu, Liang, & Gao, 2011; Kempe, Kleinberg, & Tardos, 2005; Kim & Srivastava, 2007; Stephen & Toubia, 2009). Most of these studies analyzed the advertisement value (aka influence) of a user on his friends. However, knowing that word-of-mouth is not distributed in the absence of propagation, only a few studies investigated the information propagation and its patterns. Meanwhile, some papers relied on the results of studies on propagation in offline social networks (Van den Bulte & Joshi, 2007), but these results are not necessarily valid in online environments (Howison, Wiggins, & Crowston, 2011). In general, while the cascade of information in social networking websites is generally observed, there is little data available on viral propagation in the online world, and studies on how and why the propagation occurs have received little focus. At the same time, little attention has been dedicated to measuring and characterizing the propagation of information in online social networks.

As mentioned earlier, online social networks are different and most probably follow different information dissemination patterns compared to offline social networks. Three of the major differences that affect propagation, are (a) the fact that communication can be either one-way or two-way - one-way communication is not usually seen in offline social networks; (b) due to the ease of information transmission on online networks, nodes have access to more friends instantly, so a broadcast message can easily be transmitted to many friends at once; and (c) there is more than one definition for links between nodes, as they can be defined as friendship links (those who mutually follow each other), and follower links (those who follow others without the others necessarily following them). There is, nonetheless, another important difference that deals not with the structure of the network, but with the online environment: information on online social networks is easily and readily available, whereas gathering offline social network data takes much more effort and time (Howison et al., 2011). The abundance of information gives us the opportunity to analyze online social networks for understanding the speed and magnitude of information propagation. We are also interested in evaluating the role of friends as opposed to followers in the information propagation. In order to achieve this, we evaluated information propagation on the YouTube social network. We chose YouTube because it provides the opportunity to analyze the role of friends and followers in the context of content propagation without switching between different environments and different features. The results of our study may be of interest to the online marketing community since the results may guide online marketers in choosing the more suitable social network (friendship or follower) for their viral marketing campaigns.

The rest of the paper is organized as follows. The next section reviews previous studies on content propagation. Section 3 describes the YouTube social network and its characteristics. In Section 4 we explain our data extraction method, and describe our collected data. In Section 5 we analyze the propagation magnitude on YouTube. In Section 6 we investigate the correlation of propagation and popularity. Discussion and conclusion are provided in Sections 7 and 8.

2. BACKGROUND AND MOTIVATION

There is abundant literature on the theory of propagation and new product diffusion in marketing science research (Bakshy, Karrer, & Adamic, 2009). Some models, such as the Bass model (Bass, 2004), focus solely on the behavioral aspect of propagation, and leave out the structural component of social networks. They suggest that a greater number of content generators, independent of where in the network they are located, lead to a higher propagation rate. On the other hand, models that studied the structure of social networks (Chatterjee & Eliashberg, 1990) have not been tested extensively in different networks with various structures (Bakshy et al., 2009). Therefore, those studies are still in the theoretical phase. Our current research, along with others mentioned in this section, tries to provide some empirical results on top of the theory to help in understanding social propagation and developing more realistic theoretical models.

Among the few studies that had focused on propagation patterns in social networks is the study of Flickr to measure the propagation of photos (Cha, Mislove, & Gummadi, 2009). The authors collected and analyzed a large longitude of user interactions on Flickr. They found that popularity has a loose relationship with propagation, as the popular photos were not propagated more than an average of three hops in the social network. They also observed that the propagation takes a much longer time than what is expected by marketing research. However, they could conclude that more interaction accounts for an important factor in the extent of propagation and can also expedite the propagation process. However, the study only considers friendship relations (two-way relations), and leaves out follower relations (one-way relations), which are found in abundance in online social networks.

In a different study (Huberman, Romero, & Wu, 2008), the dissemination of information on the Twitter social network is analyzed from a friendship point of view. The authors analyzed a large dataset of Twitter data to find posts, friendships, and interactions among users. The findings show a strong relationship between the number of posts and the number of followers meaning that more followers result in more encouragement for posting. However, the number of posts eventually saturates. The interesting finding is that in case of friends (who have a two-way relationship), the number of posts follows the same pattern, but it never saturates. It is also important to note that the number of friends may saturate, but the number of followers may grow indefinitely. Therefore, the authors conclude that the visible network of interactions is not the true representative of the actual hidden network that influences the propagation.

In another study, Yoganarasimhan (Yoganarasimhan, 2010) studied the effects of network structure on propagation, and discovered that in addition to the effect of neighbors' behavior, the size and structure of the initiator's network (initiator here means initiator of the content) have a great effect on the magnitude of propagation. The author calculated the centrality metrics for YouTube users, and discovered that as the size of a community increases, the central nodes of that community have a better chance of propagating their videos.

On the other hand, Cheng et al. (Xu Cheng, Dale, & Jiangchuan Liu, 2008) evaluated the relationships between YouTube videos. According to the authors, the statistics related to YouTube videos are very different from statistics of other video sharing websites. They related these differences to the social nature of YouTube, and their analysis of relations between YouTube videos confirmed this hypothesis by showing a "small world" network between YouTube videos. Although Cheng et al. evaluated the role of each YouTube video in the propagation of similar videos; they did not investigate the role of the underlying social network of video uploaders (initiators) in this propagation.

Baluja et al. (Baluja et al., 2008) conducted a similar study but took a different direction. They developed an algorithm to facilitate the propagation of preferences in social communities. They applied their algorithm on YouTube considering it a network of videos. They, therefore, reached similar results as Cheng et al. (Xu Cheng et al., 2008), confirming that YouTube's graph of videos is an effective system to propagate videos online. But they did not measure the effects of user social networks on video propagation.

On the other hand, studies such as the one by Lange (Lange, 2007) pointed out the importance of user social networks on YouTube video propagation. The author extracted user behaviors from the network of users based on video preferences, and discovered a similarity of behavior among friends, and suggested a potential for propagation in such social networks.

3. THE YOUTUBE SOCIAL NETWORK

YouTube, a subsidiary of Google, is the largest video sharing website containing about 43% of all videos found on the Internet (comScore 2010). Since its launch in 2005, the popularity of YouTube has consistantly increased, and more web users, from various demographics, registered on this video sharing website to benefit from its contents and features. Statistics from 2010 state that more than 35 hours of video are uploaded to YouTube every minute (YouTube LLC., 2010). But YouTube is not just a video sharing website. It also accounts for being a social network since it has a large number of registered users (i.e., channels) who can upload videos, follow other channels (i.e., subscribe),

Table 1. Statistics of collected data for friendship network

Data Description	*Statistics*
#Users	8,984
#Videos	113,562
#Friendship-links	8,986

Table 2. Statistics for collected data for subscribers' network

Data Description	*Statistics*
#Users	9,633
#Videos	332,296
#Subscription-links	40,358

and be friends with other channels. Thus, there are many channels in YouTube with millions of friends and subscribers (YouTube LLC., 2010). Most importantly, in order to fully qualify as a social network, YouTube has to enable users to communicate with each other. YouTube satisfies this requirement by implementing a broad infrastructure that allows users to communicate with each other in many different ways which resulted in users commenting on nearly 50% of YouTube videos (YouTube LLC., 2010). YouTube's communication infrastructure includes the following features:

- Private messaging: channels can send private messages to each other
- Commenting on channels: channels can comment on other channels
- Commenting on videos: channels can comment on videos posted by themselves or other channels
- Marking a video as favorite (favorite marking): channels can favorite uploaded videos
- Publishing video descriptions: the uploader channel can write a video description for its uploads
- Liking or disliking a video description or a comment (rating): channels can like or dislike video descriptions or comments that are posted by other channels
- Replying to a comment: every channel can reply to a comment. This is simply the act of commenting on comments.

YouTube provides the advantage of allowing two types of relationships between channels: friendship, which creates a two-way relationship for channels, and subscription, which allows channels to get updates on any other channel while having a one-way relationship with those channels. This feature allows us to evaluate our model on friendship and subscription on the same social network with the same communication features. Note that since private messages are not extractable, from an external observer's view point, the communication features are the same for both friends and subscribers. The existence of this feature is very important as it gives the opportunity to analyze the behavior and communication patterns of friends and subscribers, as well as their influence on content propagation.

4. DATA COLLECTION

Google (YouTube owner) published a library of APIs and tools that enable developers to connect their applications with Google products. APIs are a set of message formats that facilitate communication between different applications. In order to collect data we used YouTube APIs (http://code.google.com/apis/youtube/overview.html), and crawled a subset of the YouTube network. We randomly selected a YouTube video and chose its uploader as our starting point. In addition to recording all publicly available communications, uploads, and their information, we located the uploader's friends and subscribers. We continued crawling by performing the same tasks for the friends and subscribers. Note that we conducted this operation separately for friends and subscribers, as each has its own network hierarchy. In this way, only for the friendship network, we collected a subset of 9000 users, which resulted in data on 110 thousand videos and 16 million interactions in a snowball sampling method. We should mention that we collected the interactions as signs of content propagation because YouTube has a system that reveals recent activities of friends and subscribers, so every comment is visible to all neighboring nodes. We did not evaluate the content of comments, so spam might be among our collected data. However, considering that we are mainly interested in comments made by friends or subscribers or their networks, the amount of spam can be small compared to meaningful comments, the small error created by spam can be ignored. Table 1 and Table 2 contain the statistics of our collected data.

YouTube Statistics

Analysis of the extracted network of YouTube (from this point on, we refer to the extracted subset of YouTube as simply YouTube network) users shows that with the extraction of about

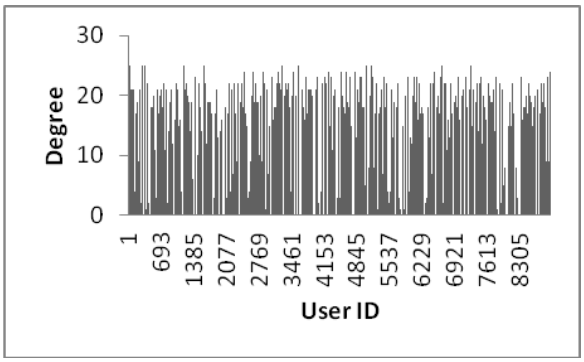


Figure 9. Degree distribution in friendship network

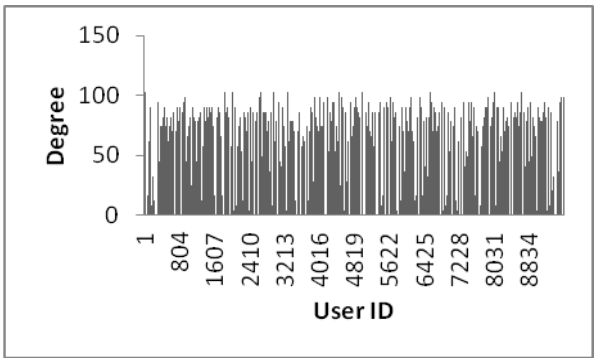


Figure 10. Degree distribution in subscription network

9000 friends using snowball sampling, we reached a maximum of 5 hops from the seed user. This shows the connectedness rate in the friendship network in the YouTube social network. Each user has an average degree of 2, with variance of 16.28, and 8301 users having only one friend, and the highest number of friends for a user in our sample is 26 (Figure 9). These statistics mean that users tend to have a small number of friends on YouTube.

On the other hand, statistics for the subscription network are different. Every user is subscribed to an average of 7 channels, with a maximum of 103 subscriptions (Figure 10). However, the number of users with zero subscription is still high and is equal to 3046. This means that the ease of subscription and lack of necessity to be approved by the other user are factors that encourage users to subscribe to other channels rather than create a friendship link. These statistics help us understand the underlying network structure of the crawled data.

Table 3. Video propagation Methods in YouTube

Propagation Method	*Description*
Sharing	Users can share YouTube videos by email, posting on blog, etc.
Recommended Videos	Videos that are recommended by YouTube based on user's previous visits.
Featuring	YouTube features some videos on its first page.
Suggested Videos	Videos that are similar to the video that the user is watching
Search Results	Videos that appear in search results
Recent Activities	Videos that were involved in recent activities of user's

Figure 9 and Figure 10 reveal an interesting fact about the networks of friendship and subscription. On the charts, the two networks seem to have similar distributions. We normalized the variances of both datasets, and the close values of variance (18.54 and 16.28) confirm this observation. Therefore, without considering the type of social network, the distribution of links follows a similar trend.

Limitations in data collection

Unfortunately, YouTube does not keep track of more than 7500 comments for each video, so we could not evaluate the speed of propagation. However, the most popular video was uploaded in 2006, and still receives comments. All the first thousand popular videos received their last comment on the day of data collection in 2011.

Moreover, this limitation may affect our results if friends and subscribers were among the people who commented first on the videos. To measure this effect, we selected a smaller dataset of videos with less than 7500 comments and ran the analysis on them. Our analysis, nevertheless, showed similar results on propagation magnitude, and its correlation with popularity.

5. PROPAGATION IN YOUTUBE

YouTube data can be propagated by different means, and is not restricted to commenting inside the YouTube network. These methods range from inside network propagation to exporting the video on a personal blog or website. Table 3 provides a set of methods that contribute to content propagation in YouTube.

Table 4. Propagation of videos in friendship network

Propagation Magnitude	*#Videos*	*%Propagated Videos*	*%Total Videos*
1 hop	1289	96.84%	1.14%
2 hop	40	3.00%	0.04%
3 hop	2	0.16%	0.01%

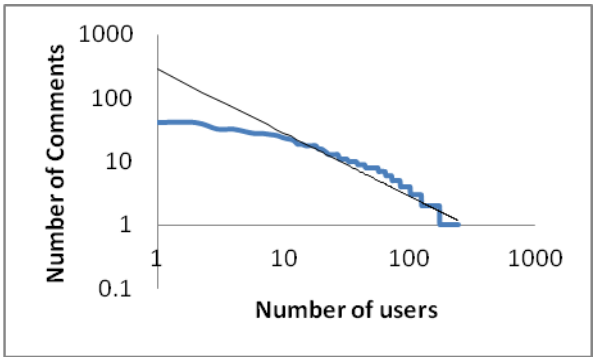


Figure 11. Distribution of comments per

Since we are interested in content propagation on YouTube that is generated by friends or subscribers, we are interested in the users' recent activities (i.e., five most recent uploads, commenting, rating, etc. that appear on every user's profile page) that are visible to friends and subscribers. Rating, favorite marking, Commenting on a video, and uploading a new video are the commonly observed recent activities, with rating being the most common one. As YouTube does not allow access to ratings or favorite markings per user, we only extracted the networks of users who commented on each others' videos. These networks include data on comments that are made on videos by users who have a path through friendship or subscription to the uploader. In other words, we eliminated from our analysis comments that were not made by friends, subscribers, and their networks.

Propagation Magnitude in YouTube

The first step in analyzing the propagation is to analyze the magnitude, or the longest hop, by which data propagates. Our dataset of 5 hops shows interesting results. We discuss them in the friendship and subscription datasets.

Propagation Magnitude in Friendship Network

We recorded a total 16.4 million interactions on videos that are posted in our friendship dataset.

Table 5. Propagation of videos in subscription network

Propagation Magnitude	*#Videos*	*%Propagated Videos*	*%Total Videos*
1 hop	269	96.76%	0.88%
2 hop	9	3.24%	0.03%

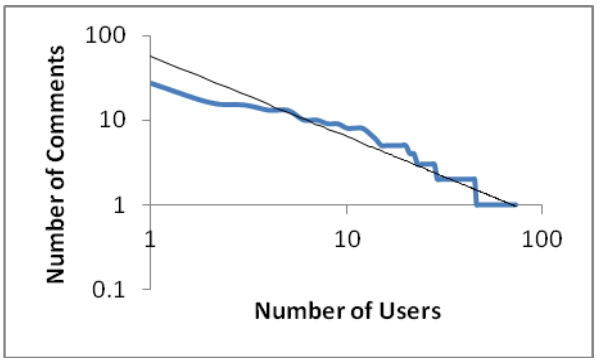


Figure 12. Distribution of comments in

Since we are only interested in interactions between friends, we pre-processed our data to extract the underlying network of interactions between friends. This resulted in a huge reduction in our sample graph. This illustrates our first finding: in an open social network, the amount of interactions between strangers accounts for a high percentage of the total interactions.

This finding is verified by a reduction of our captured interactions to 133 thousand interactions, a reduction rate of 98.76%, when we filtered out the interactions between channels that do not have a friendship path to the uploader node.

Analysis of propagation in the friendship network revealed that videos are propagated at most to three hops of friends (a hop denotes a link between two levels of friendship). Meanwhile, the distribution of propagation reveals that only a small fraction of the videos is propagated to the second and third levels of friends (Table 4). The propagation of videos through friendship is not significant. However, looking at the users involved in propagating the videos suggests that a huge part of propagation is carried out by a small number of users. We observed that the commenting pattern in the friendship network follows a power law distribution with the exponent of 0.90, meaning that the contents are highly propagated through a small number of highly active users (Figure 11).

Propagation Magnitude in Subscription Network
In the same way, we recorded a total 44.7 million interactions on videos that are posted in our subscription dataset. Since we are only interested in interactions between subscribers, we pre-processed our data to extract only the interactions between subscribers. Similar to the friendship network, this resulted in a huge reduction in our sample graph. The captured interactions were reduced to 27 thousand, much less than the interactions in the friendship network. This reduction has a rate of 99.93%, which means that almost all interactions happen between users who do not have a path through subscription. This was a surprise because since the connectedness of the subscription network is far higher than the friendship network, it was expected that subscribers have more effect on propagation than friends. The low effect on propagation may be due to lower personal connection between subscribers, hence subscribers are less inclined to leave comments. Meanwhile, our analysis of propagation in the subscription network revealed that videos are propagated at most to two hops of subscribers. Moreover, the distribution of propagation suggests that only a small fraction of the videos are propagated to the second level of subscribers.

Similar to the friendship network, the propagation of videos through subscription is not significant. However, looking at the users who are involved in propagating the videos still suggests that a huge part of propagation is carried out by a small number of subscribers. We observed that the commenting pattern in the subscription network follows a power law distribution with the exponent of 0.93, meaning that the content is highly propagated through a small number of highly active users (Figure 12).

6. PROPAGATION AND POPULARITY

In the next step, we investigated the popularity of videos in relation to their propagation, in order to understand whether the popularity of videos drives or is driven by propagation, or if friends and subscribers choose the videos to comment on based on other considerations. To do so, we selected a set of ten highly propagated videos in addition to ten highly popular videos from each dataset, and evaluated the correlation of popularity and propagation of videos. We measure the popularity of a video by its view count and ratings. Table 6 shows statistics of the five most popular videos in our datasets. These videos may or may not be propagated by network members, and these statistics show general popularities of videos without considering their propagation. Note that three of five popular videos are common in both networks. This infers the similarity of growth patterns in both networks.

Propagation and popularity in friendship network

To measure the correlation between popularity and propagation in the friendship network, we extracted the five most popular and the five longest propagated nodes from the network of friendship interactions, i.e., the friends who commented on each other's posts (Table 7). In our first observation, none of the videos that appeared in the network's most popular videos (Table 6) appeared in the most popular and deepest propagated set in the friendship interaction network, and the most popular video in the friendship interaction network was, in

Table 6. Statistics of popular videos

Dataset	*View Count*	*Rating*
Friendship	$1.8 * 10^8$	4.68
	$8.6 * 10^7$	4.91
	$4.8 * 10^7$	4.83
	$4.6 * 10^7$	4.54
	$3.8 * 10^7$	4.93
Subscription	$1.8 * 10^8$	4.68
	$4.8 * 10^7$	4.83
	$3.8 * 10^7$	4.93
	$3.4 * 10^7$	4.91
	$3.6 * 10^7$	4.50

Table 7. The deepest propagated, and the most popular videos in friendship network

Type	*Propagation Depth (hops)*	*View Count*	*Rating*
Longest Propagated	3	575	5
	3	231	3.67
	2	71953	3.78
	2	61429	3.95
	2	30914	4.75
Most Popular	1	562261	4.94
	1	558523	4.89
	1	78220	4.94
	1	78074	4.93
	1	76163	4.39

Table 8. The deepest propagated, and the most popular videos in subscription network

Type	*Propagation Depth (hops)*	*View Count*	*Rating*
Longest Propagated	2	72001	3.78
	2	61429	3.95
	2	30935	4.75
	2	7262	4.43
	2	5072	3.96
Most Popular	1	562611	4.94
	2	72001	3.78
	2	61429	3.95
	1	37201	4.82
	2	30935	4.75

fact, ranked 1570 out of 113 thousand videos in the total friendship network. Meanwhile, the longest propagated videos had average popularities in the friendship network. These figures mean that the propagation of videos by friends does not affect the popularity of videos, and vice versa.

Propagation and popularity in subscription network

We applied the methodology that we used for the friendship network on the subscription network. The analysis of the subscription network shows that the most popular video (Table 8) ranked 747 out of 332 thousand videos in the total subscription network (Table 6). On the other hand, videos that are propagated the most in the subscription network are also subscription network's most popular videos. Therefore, there is a correlation between the popularity and the level of propagation by subscribers, meaning that more propagated videos by subscribers become popular at least among the subscribers and their network or vice versa.

7. DISCUSSION

Advertisement is a costly process for businesses, and in some cases, it takes a considerable amount of the business budget. Businesses have always looked into ways to advertise their products and services at a lower cost. Viral marketing and advertisement on social networks provided a solution for this requirement. However, there is still a considerable cost associated with viral marketing even though it is lower than, say, banner ads. This cost is mainly associated with influencing the first person and encouraging him/her to spread the word, in addition to making sure that the word will be spread to the next levels in the network. Therefore, businesses may be interested in finding the most appropriate person and the most appropriate network to do the advertisement. The low propagation rate among friends and followers in an open social network suggests that open social networks are not generally well suited for businesses that need to spread the word in communities. Meanwhile, the better propagation rate among friends (compared to followers) suggests that the focus of businesses should be on friendship networks. At the same time, our research suggests that in friendship networks, the popularity of the message does not affect its propagation, while in follower networks it does. Therefore, businesses may need to focus on making the message itself interesting (popular) within follower networks more than they do within friendship networks.

8. CONCLUSION

There are many studies on the effects of social networks on viral marketing and diffusion of information. However, few studies have focused on propagation in social networks. Moreover, to the best of our knowledge, there is no study that analyses propagation in friendship and follower networks as two different entities in the same environment, and at the same time. Therefore, we felt a need for a study of propagation, its trends, and magnitude. We conducted a propagation analysis on an open social network, i.e., YouTube. We believe that the fact that everyone can view the contents uploaded by a user, and post a comment, rate, or share that content contribute to YouTube's openness.

We crawled two subsets of the YouTube user network for friendship and subscription and analyzed the propagation, and the role of friends and subscribers in content dissemination. We observed that the effect on propagation of people who are not either in a friendship network or a subscription network is higher than that of friends or subscribers. Meanwhile, we discovered that even though the network of subscribers was denser than the network of friends, the propagation in the subscription network was lower. This might imply that when the relationship is one-way, users are less inclined to contribute to the content.

Although our extracted data did not initially include user relations to the level of more than 5 hops, this limitation did not affect our study of the magnitude of propagation, and the correlation of propagation and popularity as even the most popular videos did not propagate more than three hops in their networks. Our result shows a low correlation between popularity and propagation in general. However, the correlation of popularity and propagation in the friendship network is more than what exists in the subscription network. This may be due to the fact that friends feel more obliged than subscribers to contribute comments about the contents posted by their peers. On the other hand, subscribers may, most of the time, only comment on what interests them.

As future work, we intend to extract a larger dataset, and combine the networks that are common in both friendship and subscription datasets in order to analyze the effects of propagation in the existence of both friends and followers at the same time. Analyzing the resulting network will lead to a better understanding of social networks as a marketing channel, and will lead marketers to a more suitable choice of network for their marketing campaigns.

9. ACKNOWLEDGEMENT

We would like to acknowledge Mr. Hamid Poursepanj for his invaluable insights and efforts that were put in this project.

10. REFERENCES

Asur, S., & Huberman, B. A. (2010). Predicting the Future with Social Media. SSRN eLibrary. Retrieved from http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1579522

Bakshy, E., Karrer, B., & Adamic, L. A. (2009). Social influence and the diffusion of user-created content (p. 325). ACM Press. doi:10.1145/1566374.1566421

Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnik, J., Kumar, S., Ravichandran, D., et al. (2008). Video suggestion and discovery for youtube: taking random walks through the view graph. Proceeding of the 17th international conference on World Wide Web, WWW '08 (pp. 895–904). Beijing, China: ACM. doi:10.1145/1367497.1367618

Bass, F. M. (2004). Comments on "A New Product Growth for Model Consumer Durables": The Bass Model. Management Science, 50(12), 1833-1840.

Bearden, W. O., Calcich, S. E., Netemeyer, R., & Teel, J. E. (1986). An Exploratory Investigation Of Consumer Innovativeness And Interpersonal Influences. Advances in Consumer Research, 13(1), 77-82.

Cha, M., Mislove, A., & Gummadi, K. P. (2009). A measurement-driven analysis of information propagation in the flickr social network. Proceedings of the 18th international conference on World wide web, WWW '09 (pp. 721–730). New York, NY, USA: ACM. doi:10.1145/1526709.1526806

Chatterjee, R., & Eliashberg, J. (1990). The Innovation Diffusion Process in a Heterogeneous Population: A Micromodeling Approach. Management Science, 36(9), 1057-1079.

Domingos, P., & Richardson, M. (2001). Mining the network value of customers. Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '01 (pp. 57–66). New York, NY, USA: ACM. doi:10.1145/502512.502525

Duan, W., Gu, B., & Whinston, A. B. (2008). Do online reviews matter? - An empirical investigation of panel data. Decis. Support Syst., 45(4), 1007-1016.

Evans, D. C. (2009, January 15). Beyond Influencers: Social Network Properties and Viral Marketing. Psychster Inc. Retrieved from http://www.slideshare.net/idealistdave/beyond-influencers-social-network-properties-and-viral-marketing-presentation

Facebook Inc. (2011). Facebook Statistics. Retrieved July 9, 2011, from http://www.facebook.com/press/info.php?statistics

Howison, J., Wiggins, A., & Crowston, K. (2011). Validity Issues in the Use of Social Network Analysis for the Study of Online Communities. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.157.9237

Hu, N., Tian, G., Liu, L., Liang, B., & Gao, Y. (2011). Do Links Matter? An Investigation

of the Impact of Consumer Feedback, Recommendation Networks, and Price Bundling on Sales. Engineering Management, IEEE Transactions on, PP(99), 1-12. doi:10.1109/TEM.2010.2064318

Huberman, B. A., Romero, D. M., & Wu, F. (2008). Social networks that matter: Twitter under the microscope. 0812.1045. Retrieved from http://arxiv.org/abs/0812.1045

Kempe, D., Kleinberg, J., & Tardos, É. (2005). Influential Nodes in a Diffusion Model for Social Networks. Automata, Languages and Programming (pp. 1127-1138). Retrieved from http://dx.doi.org/10.1007/11523468_91

Kim, Y. A., & Srivastava, J. (2007). Impact of social influence in e-commerce decision making. Proceedings of the ninth international conference on Electronic commerce (pp. 293-302). Minneapolis, MN, USA: ACM. doi:10.1145/1282100.1282157

Lange, P. G. (2007). Publicly Private and Privately Public: Social Networking on YouTube. Journal of Computer-Mediated Communication, 13(1), 361-380. doi:10.1111/j.1083-6101.2007.00400.x

Leskovec, J., Adamic, L. A., & Huberman, B. A. (2007). The dynamics of viral marketing. ACM Trans. Web, 1(1), 5+. doi:10.1145/1232722.1232727

Stephen, A. T., & Toubia, O. (2009). Deriving Value from Social Commerce Networks. SSRN eLibrary. Retrieved from http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1150995

Trusov, M., Bodapati, A. V., & Bucklin, R. E. (2010). Determining Influential Users in Internet Social Networks. Journal of Marketing Research, XLVII(August), 643-658. doi:10.2139/ssrn.1479689

Van den Bulte, C., & Joshi, Y. V. (2007). New Product Diffusion with Influentials and Imitators. MARKETING SCIENCE, 26(3), 400-421. doi:10.1287/mksc.1060.0224

Xu Cheng, Dale, C., & Jiangchuan Liu. (2008). Statistics and Social Network of YouTube Videos. 16th International Workshop on Quality of Service, 2008. IWQoS 2008 (pp. 229-238). Presented at the 16th International Workshop on Quality of Service, 2008. IWQoS 2008, IEEE. doi:10.1109/IWQOS.2008.32

Yoganarasimhan, H. (2010). Impact of Social Network Structure on Content Propagation: A Study using YouTube Data. University of California, Davis.

YouTube LLC. (2010). YouTube - Press Statistics. Retrieved July 9, 2011, from http://www.youtube.com/t/press_statistics

Editor's Note:

This paper was selected for inclusion in the journal as a CONISAR 2011 Meritorious Paper. The acceptance rate is typically 15% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2011.

Maximizing Visibility in Skylines

Muhammed Miah
mmiah@suno.edu
Management Information Systems Department
Southern University at New Orleans
New Orleans, LA 70126, USA

Abstract

Given a new product (a tuple), the research in this paper considers the problem of selecting a small subset of attributes to highlight such that the product stands out in a crowd of existing competitive products and is widely visible to the pool of potential customers. This problem has applications in marketing and product manufacturing and has been the subject of recent investigations. This research considers an important variant where a product is considered to be visible to a customer if it occurs in the skyline of the query posed by the customer. Given a set of *d*-dimensional points, a skyline query returns points that are not dominated by any other point on all dimensions. This problem variant poses new challenges that cannot be solved optimally using prior techniques. In this work, we develop a novel optimal algorithm based on the Signature Tree data structure as well as approximation algorithms to solve the problem. We conduct a performance study illustrating the benefits of our methods on real as well as synthetic data.

Keywords: maximize visibility, subset of attributes, skylines, signature tree, algorithms.

1. INTRODUCTION

Skyline query processing has been extensively investigated in recent years. Given a set of points, the skyline comprises of the points that are not dominated by other points. A point dominates another point if it is as good or better in all dimensions and better in at least one dimension. For example, a student attending a conference might want to search within a hotels database for a cheap hotel with reasonable ratings near the conference venue. This kind of query sometimes contains conflicting goals, as hotels near the conference venue with reasonable ratings are expected to be rather expensive. It is of interest to return as query results the set of skyline hotels; where for each skyline hotel there is no other hotel that is cheaper, nearer, and with better ratings. While skylines are naturally defined for numeric data, they can also be defined for categorical and Boolean data if the data values within each attribute's domain have natural total (or even partial) orderings. In this paper we mainly consider Boolean skylines (skylines with Boolean data), where all the attributes asked by a query need not to be present in the tuple to be returned by the query. For example, let us consider a car database with Boolean attributes such as whether the car has *AC*, *Power Doors*, *Power Brakes*, etc. Thus if a user poses a query such as "Select * from Cars where Make = Honda and AC = yes and Power Windows = yes", then a car such as <*Toyota, AC, Power Windows*> would appear in the skyline if there is no car that exactly satisfies the query conditions.

Thus, skyline query processing techniques or skyline operators are designed to provide all interesting answers that may satisfy a user's need. Skyline semantics are less rigid than conjunctive range query semantics and can be of use in exploratory search applications.

Our goal in this paper is however not to design a skyline operator. Instead, we consider an interesting generalization of a problem that has applications in marketing and product manufacturing, and has been the subject of recent investigations (Miah, Das, Hristidis, & Mannila, 2009). Given a new product (a tuple), we consider the problem of selecting a small subset of attributes to highlight such that the product stands out in a crowd of existing competitive products and is widely visible to the pool of potential customers. So the goal is not to develop a better search technique to help the user (buyer of a product) but to help the seller of the product to reach maximum number of users. This problem was investigated by Miah et al. (2009), where primarily a somewhat rigid model of conjunctive query semantics was used to define product visibility – a product is visible to a customer if it satisfies all conditions of the query posed by the customer. The skyline variant of the problem was also discussed very briefly and the same solution was proposed for both the Boolean and skyline problem variants, which later proved not to be optimal for skyline variant (Miah, 2009). In this paper we mainly consider skyline semantics – a product is visible to a customer if it appears in the skyline of the customer's query, i.e., although it may not exactly match all query conditions, it is nevertheless potentially "more interesting" to the customer than many other competing products.

Selecting [a/the] subset of attributes to highlight a product plays an important role in marketing and manufacturing of products as well as in operation[s] research. We can consider a real world scenario: assume that one wishes to publish a classified-ad in a newspaper (online or printed) to advertise a house for sale. The house may have a lot of attributes (number of bedrooms, bathrooms, close to beach, close to school, etc.). However, due to the advertisement costs involved, it is not possible to describe all attributes in the ad. So one has to select, say the ten best attributes. Which ones should be selected within the budget limit such that the published ad will be viewed by as many customers as possible? From a manufacturing point of view, a house builder might want to find the most popular combination of features to add to a house to be constructed in so as to be more interesting than other competing homes to as many potential customers as possible. Another interesting real world example would be to select a small set of keywords or a title for an advertisement campaign.

In this paper, we mainly focus on the important and interesting variant of the problem where the data is Boolean and the queries follow skyline retrieval semantics. Here, a tuple does not have to have all the attributes present asked by a query, but it has to be visible on the skyline of the query. This problem variant cannot be solved optimally by the existing algorithms. Moreover one query can have multiple skylines, i.e., multiple sets of attributes for which different data points (tuples) are visible on the skyline of the query. We develop a technique to solve the skyline version of the problem that is quite different from the methods proposed by Miah et al. (2009). The new method is based on a judicious application of the signature tree data structure (Chen & Chen, 2006 and Miah, 2009) with modifications and smart prunings. Interestingly, our new algorithm can also solve easily the old problem variant of conjunctive query semantics optimally.

Our main contributions are summarized below:

1. We investigate the problem of selecting attributes of a tuple for maximum visibility in skylines as a promising data exploration problem that benefits a certain class of users interested in designing and marketing their products.
2. Though the problem is proved to be NP-hard (Miah et al., 2009), we are able to develop an optimal algorithm based on the signature tree data structure to solve the problem that works well for moderate problem instances.
3. We also present fast approximation algorithms that work well for larger problem instances.
4. We perform detailed performance evaluations on both real and synthetic data to demonstrate the effectiveness of our developed algorithms.

2. PROBLEM FRAMEWORK

Before formally defining the problem, we first provide some useful definitions and notations in Appendix 1.

The problem formally can be defined as follows.

PROBLEM: *Given a database of competing products D, a query log Q with Skyline Query semantics, a new tuple t, and an integer m,*

compute a compressed tuple t′ by retaining m attributes such that the number of queries that retrieve t′ on the skylines is maximized.

Car ID	*Attributes/Features present in the car*
t_1	{AC, Four Door, Power Doors}
t_2	{Four Door, Turbo}
t_3	{AC, Auto Trans, Power Brakes, Power Doors}
t_4	{AC, Four Door, Power Brakes, Power Doors}
t_5	{AC, Four Door}
t_6	{Four Door, Power Doors}
t_7	{Power Doors, Turbo}

Database *D*

Query ID	*Attributes/Features asked by the query*
q_1	{AC, Four Door}
q_2	{AC, Power Doors}
q_3	{Four Door, Power Doors}
q_4	{Power Brakes, Power Doors}
q_5	{Auto Trans, Turbo}

Query Log *Q*

New Car	*Attributes/Features present in the car*
t	{AC, Auto Trans, Four Door, Power Brakes, Power Doors}

New tuple *t* to be inserted

Figure 2. Running EXAMPLE 1

The following running example will be used to illustrate problem.

EXAMPLE 1: *Consider an inventory database of an auto dealer, which contains a single database table D with N=7 rows and M=6 possible attributes a car can have (AC, Auto Trans, Four Door, Power Brakes, Power Doors, and Turbo) where each tuple represents a car for sale. The table has numerous attributes that describe details of the car: Boolean attributes such as AC, Four Door, etc; categorical attributes such as Make, Color, etc; numeric attributes such as Price, Age, etc; and text attributes such as Reviews, Accident History, and so on. Figure 2 illustrates such a database (where only the Boolean attributes are shown) of seven cars already advertised for sale. The figure also illustrates a query log of five queries, and a new car t that needs to be advertised, i.e., inserted into this database.* □

Now we can find the skyline points (cars) for the given database *D* and Query log *Q* in Figure 2. As we know, a skyline point is a point which is not dominated by any other point in all dimensions. For query q_1 {*AC, Four Door*}, we can see it easily that tuples t_4 and t_5 are the tuples (skyline points) which are not dominated by any other tuples in *D*. For query q_2 {*AC, Power Doors*}, tuples t_3 and t_4 are the skyline points, and so on. Table 1 (Appendix 2) displays all the skylines found for the given query log *Q* and database *D*. A skyline tuple or data points can have many attributes but we are interested only in the attributes for which the tuple is visible on the skyline, as our goal is to find the subset of attributes for the new tuple which will maximize the number of queries having the new tuple visible on their skylines. A query can have more than one skyline; e.g., for query q_5, tuples t_2 and t_7 are visible on one skyline whereas tuple t_3 is visible on another skyline. We keep separate record for each skyline as shown in Table 1 (Appendix 2).

Suppose we are required to retain $m = 3$ attributes of the new tuple. It is not hard to see that if we retain the attributes *AC*, *Four Door*, and *Power Doors* (i.e., *t′* = {*AC*, *Four Door*, *Power Doors}*), the compressed tuple *t′* will be visible on the skylines for the maximum of three queries (q_1, q_2, and q_3). No other selection of three attributes of the new tuple will remain on skylines of more queries.

3. OPTIMAL ALGORITHM

There are several methods proposed for efficient processing of skyline queries such as Block-Nested-Loops and Divide & Conquer (Kossmann & Stocker, 2001), Bitmap-based and Index-based (Tan et al. 2001), Nearest Neighbor (Kossmann, Ramsak, & Rost, 2002), Branch and Bound Skyline (Papadias, Tao, Fu, & Seeger, 2003). Any good skyline processing technique can be used here to find the skylines for the query log. We can assume that the skylines for each query in the log have already been computed by any one of these algorithms. Once these skylines have been found, then our problem is to find the subset of the attributes for the new tuple so that skylines from the

maximum number of queries will retrieve the new tuple.

A Naïve optimal algorithm and its infeasibility are discussed in Appendix 3.

We propose a novel optimal algorithm based on Signature Tree data structure (Chen & Chen, 2006) which is much more efficient than the Naïve algorithm.

Optimal Algorithm Based on Signature Tree (*AST*)

We adapt the candidate set generating function *apriori-gen* used in the *Apriori* algorithm for mining association rules (Agrawal & Srikant, 1994) to generate possible candidate sets in this algorithm. The *apriori-gen* function takes L_{k-1}, the set of all large (k-1) itemsets. It returns a superset of set of all large k-itemsets. The function works as shown in Figure 3.

First in the *join* step, it joins L_{k-1} with L_{k-1}:

Insert into C_k
select $p.\text{item}_1$, $p.\text{item}_2$, …, $p.\text{item}_{k-1}$, $q.\text{item}_{k-1}$
from L_{k-1} p, L_{k-1} q
where $p.\text{item}_1 = q.\text{item}_1$, …, $p.\text{item}_{k-2} = q.\text{item}_{k-2}$,
$p.\text{item}_{k-1}$, = $q.\text{item}_{k-1}$

Next, in the *prune* step, it deletes all itemsets $c \in C_k$ such that some (k-1) subset of c is not in L_{k-1}:

for all itemsets $c \in C_k$ do
for all (k-1)-subsets u of c do
if ($u \notin L_{k-1}$ then
delete c from C_k

Figure 3. Function *apriori-gen*

The signature tree, its construction, and definition are discussed in Appendix 4. We build a balanced signature tree for the skylines using the weight based method (Chen & Chen, 2006). The process of creating the balanced signature tree is discussed in Appendix 4.

The traditional approach of searching the signature tree is discussed in Appendix 5.

The traditional searching of signature tree has two major problems (discussed in Appendix 6). So we propose a new approach for searching the signature tree.

New Approach for Searching the Signature Tree: First we create the signature tree for the skylines as described above. Then, we search the tree at each level from *2*-attributes candidate sets to up to *m*-attributes candidate sets. Candidate sets at each level k (= 2…m) are generated using function *apriori-gen* as discussed above. At each level searching is done as follows:

i. Let v be the node encountered and $s_t[i]$ be the position to be checked.
ii. We move both the right and left child of v whether $s_t[i] = 0$ or $s_t[i] = 1$.
iii. We maintain a variable $m' = (m - k)$, which is the number of *mistakes* allowed. Here, m is the number of attributes we need to retain for the new tuple t, and k is the current number of attributes in the candidate set. A mistake during the search occurs when we move to the right of a node (i.e., skyline has value 1 for the digit mentioned by the node) and the candidate set has value 0 for that digit. If this situation happens, we increase the count for mistakes. Consider the signature tree in Figure 5 (Appendix 4) for our running example. Assume current value of k = 2, m = 3, and we have a candidate set with signature 110000. So, the value of m' = m - k = 1. As mentioned above, when we search the tree, we move both right and left of a node. Moving left never increases the number of mistakes because if a skyline s has value 0 for a digit then a candidate set c can have either value 0 or 1 for the corresponding digit. When we move right from the root node (Figure 5 in Appendix 4), the value of digit 5 (root node) in candidate set 110000 is 0, so we increase the number of mistakes made so far, which is (0+1) = 1. Next we move both left and right from node labeled 1 (right child of root). Moving right from node labeled 1 does not increase the number of mistakes made as the value of digit 1 in candidate set is also 1. But when we move right of the node labeled 3 (left child of node labeled 1), we increase the number of mistakes made because value of digit 3 in the candidate set is 0. Here a new value for the number of mistakes made so far is (1+1) = 2 which is greater than m'. So we do not consider any nodes to the right of node labeled 3. As we can see from the tree in Figure 5 (Appendix 4), we do not consider s_3 with signature

001010 as a possible subset of the candidate set 110000 in future. We can see easily that adding 1 to the candidate set 110000 will not make s_3 (001010) a subset of the candidate in future.

iv. Once we reach a node and number of mistakes made so far reaching the node from the root is greater than *m'*, then we do not consider the node and its children (if it is an internal node) as the possible subset of the candidate set.

v. Once we reach a leaf node (skyline) and the number of mistakes made so far reaching the node from the root is not greater than *m'*, we keep the skyline for further match.

Once we find the corresponding skylines *S* (leaf nodes) by searching the tree for a candidate set *c*, for each skyline s_i we do the following:

a) We find the number attributes *r* present in skyline s_i which is not present in candidate set *c*. If $r \leq (m - k)$, we count s_i as the possible subset of *c*. Here, *m* is the number of attributes we need to retain for the new tuple *t*, and *k* is the current number of attributes in the candidate set. If $r > (m - k)$, we do not consider that skyline or leaf node for *c* as a possible subset, i.e., do not increase count for *c*.

b) We keep a count for each candidate set *c* that how many skylines have been found as possible subsets of *c*. Here, we count each query only once. For example, considering our running example, if we find two skylines s_5 and s_6 are the subsets of any candidate set c_i, we only count them once because both come from the same query q_5. This is why we keep information in the tree for both the skyline and the query from which it came. We remove the candidate set *c* if the total count for it is less than the *minimum support*.

c) At level *m* (candidate sets with *m*-attributes), we simply check how many skylines (found after searching the tree) are actually the subsets of the candidate set. Again, we count skylines for each query only once for a candidate set. We return the candidate set with highest count as the top-*m* attributes for the new tuple *t*.

4. APPROXIMATION ALGORITHMS

A simple greedy heuristic would be to select the top-*m* attributes with highest frequency in the skyline log (frequency is the number of times an attribute appear in the skyline log). But this is not a good approach when attributes are correlated, which is quite common in practice. We propose three effective approximation algorithms based on greedy heuristics that perform well.

Backward Elimination (*BE*)

We propose a backward elimination greedy heuristic where all single attributes are considered first and then remove one at a time until *m* attributes are left. The summary of the approach is shown in Figure 6 below.

1. Take the original set of attributes, S_o.
2. Remove an attribute randomly from S_o which was not tested (removed) before and count how many skylines are subsets of the new set (S_o -1).
3. Restore the attribute removed in *step 2*.
4. Repeat *steps 2* and *3* for each attribute in S_o.
5. Remove the attribute from S_o permanently with highest count, i.e., remove the attribute which has lowest impact.
6. Repeat *steps 2-5* until S_o remains with *m* attributes

Figure 6. Approximation Algorithm: Backward Elimination (*BE*)

1. Let S_o = set of all original attributes present in skyline log
2. Let *S* = an empty set, and an integer $k = 0$ which is the number of attributes currently present in *S*.
3. For ($k = 0$ to *m*)
4. For each attribute a_i left in S_o, check if we add (m-k) attributes from S_o including a_i to *S*, then how many skylines could be possible subsets of *S*. In fact we check the number of attributes in a skyline which are not present in *S* including a_i. If the number is less than (*m* – number of attributes in *S* including a_i), then the skyline is considered as a possible subset of the candidate set *S*
5. Add the attribute to *S* with highest count in *step 4*.
6. Remove the attribute added to *S* in *step 5* from S_o.
7. End
8. Return *S*.

Figure 7. Approximation Algorithm: Forward Selection (*FS*)

Forward Selection (*FS*)

Forward selection heuristic starts with an empty set of attributes, *S* and adds one attribute at a

time until *S* has *m* attributes. In order to find the best *m* attributes, we can first add the attribute with highest frequency in the skyline log (frequency means number of times the attributes appear in the skyline log). Next we add the attribute which occurs most with the first attribute, then add the attribute which occurs most with the first and second attributes together, and so on until *S* remains with *m* attributes. In this method, we might find *S* is a good selection of *m* attributes if we want to find the new *t* as a subset of maximum number of skylines or queries. But our goal is the opposite; we want to find *S* as a superset of the maximum number of skylines. So, just adding top-*m* attributes may not result a good selection of attributes. We modify the addition criteria of an attribute to *S*. Figure 7 shows the summary of the algorithm *FS*.

Combination of Forward Selection and Backward Elimination (*FSBE*)

Now we propose another heuristic which combines both the algorithms *BE* and *FS* considering bidirectional hill climbing techniques. Hill climbing is a well known procedure for sequential attribute selection. Greedy algorithms such as *BE* and *FS* implement so called unidirectional hill climbing, i.e., attributes once added (removed) cannot be later deleted (added). The advantage of bidirectional hill climbing compared to either *FS* or *BE* is that one or several previously deleted (added) attributes can be brought back to (removed from) the subset if the accuracy of the algorithm increases. But this technique can be time consuming as both the *BE* and *FS* has to perform completely and then somehow combine the results. So, we propose a new technique to improve the performance of the algorithm, described as follows:

1. Let S_o = set of all original attributes present in skyline log
2. Let S = an empty set, and an integer $k = 0$ which is the number of attributes currently present in S.
3. For ($k = 0$ to m)
4. Perform *BE* on S_o. // every step removes one attribute from S_o.
5. Perform *BS* on S_o. // every step adds one attribute to S.
6. Remove the attribute from S_o which is added by *FS* to S in *step 5*
7. End
8. Return S.

Figure 8. Approximation Algorithm: *FSBE*

Once an attribute is removed by *BE* it is not considered to be added by *FS* anymore. Similarly, once an attribute is added by *FS* it is not considered to be removed by *BE*. So, at every step *BE* eliminates one attribute and *FS* adds one. We repeat the procedure until *FS* adds *m* attributes. The summary of the algorithm *FSBE* is given in Figure 8.

5. EXPERIMENTS

In this section we measure (a) the time cost of the proposed optimal and approximation algorithms, and (b) the quality of the approximation algorithms.

The system configuration and details of datasets are discussed in Appendix 7.

The top-*m* attributes selected by our algorithms seem very effective. For example, both for real and synthetic query logs, our optimal algorithm could select top features or attributes specific to a car, e.g., sporty features are selected for sports cars, safety features are selected for passenger sedans, and so on.

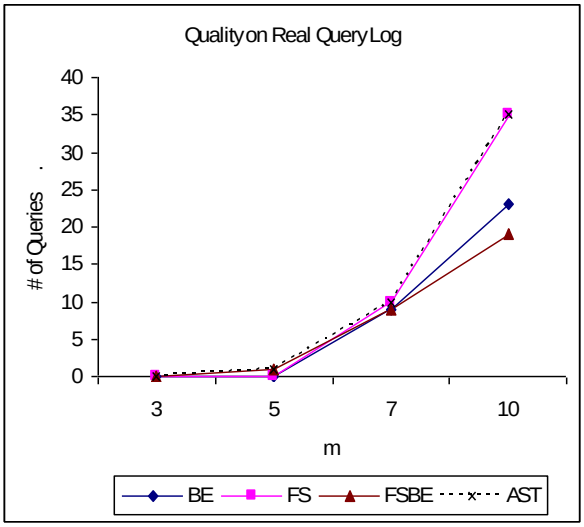


Figure 9. Quality on Real Query Log for varying *m*

Figure 9 shows the quality of the algorithms on the real query log which has a total of 185 queries. The *x*-axis represents *m* which is the number of attributes needing to be selected, and *y*-axis represents the number of queries for which the new tuple with selected *m* attributes is visible on the skylines. We use several experiments for each algorithm with varying *m*.

The real query log in fact has no query as well as skyline with less than or equal to 3 attributes, so we can see all algorithms produce zero output for m = 3. We can see from the graph that approximation algorithms work really well.

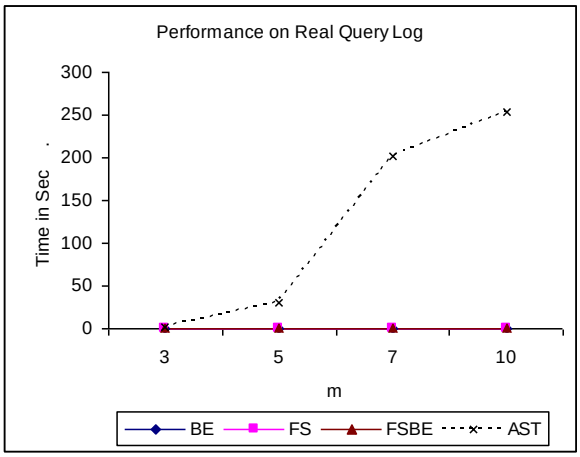


Figure 10. Performance on Real Query Log for varying m

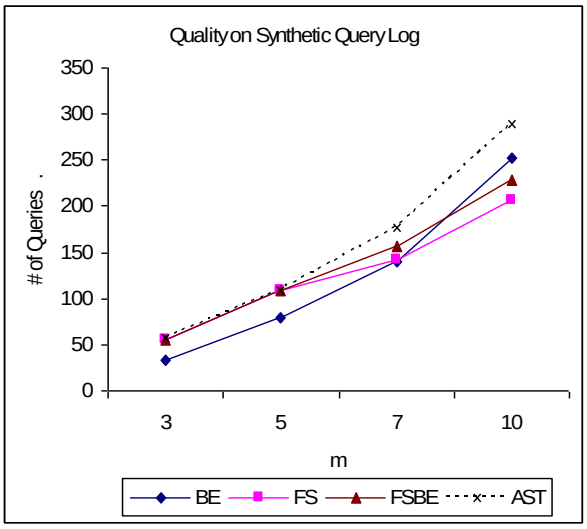


Figure 11. Quality on Synthetic Query Log (1000 queries) for varying m

Figure 10 displays the execution times of each algorithm for the real query log. Here x-axis represents m. The y-axis represents the time in seconds to execute the algorithm. We can see that the approximation algorithms are faster than optimal *AST*, which is expected.

Next we show the quality on synthetic query log of size 1000 queries in Figure 11. As we can see from the graph, for the synthetic query log our approximation algorithms also produce very good outputs similar to the real query log.

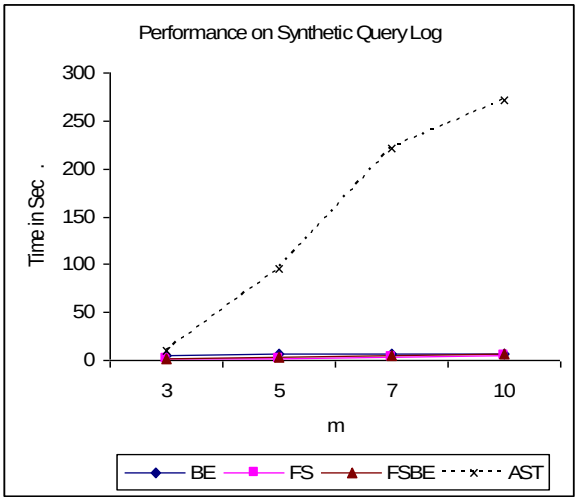


Figure 12. Performance on Synthetic Query Log (1000 queries) for varying m

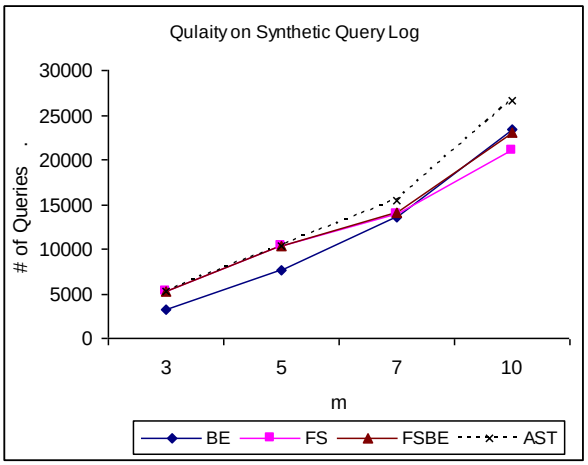


Figure 13. Quality on Synthetic Query Log (100K queries) for varying m

Figure 12 shows the execution times of the algorithms for the synthetic query log of 1000 queries. As we can see from the graph, the approximation algorithms are also very fast for synthetic query log. When m increases, execution time for *AST* also increases more than approximation algorithms.

Figure 13 and Figure 14 show the quality and execution times respectively of each algorithm for the synthetic query log of 100000 queries.

The algorithms perform similar way as in the previous cases.

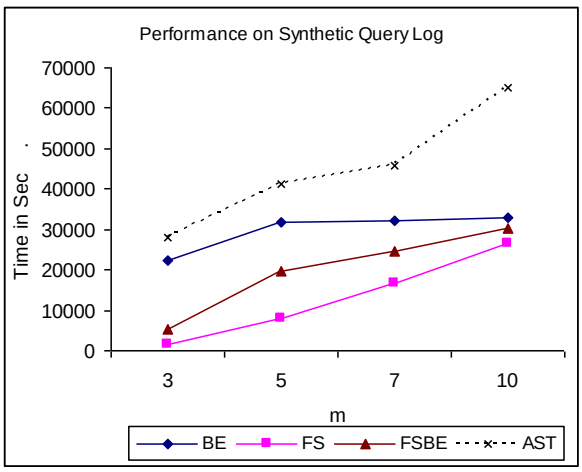


Figure 14. Performance on Synthetic Query Log (100K queries) for varying *m*

We can see from the graphs that approximation algorithm *FS* is faster than both *BE* and *FSBE*, which makes sense. *FS* starts with an empty attribute set and at each step adds one attribute until *m* attributes are added to the set. Usually *m* is a small number compared to the total number of attributes, *M* present in the database. So, *FS* has to iterate only *m* times. On the other hand, algorithm *BE* starts with a set of all *M* attributes and at each step eliminates one attribute from the set until it is left with *m* attributes. As we said, typically *M* is a larger number than *m*, so *BE* has to iterate (*M*-*m*) times which is typically much larger than *m*. So *BE* is slower than *FS*. Because *BE* has to iterate more times, it usually produces better output than *FS* that we can see easily for the synthetic query logs. *FSBE* is the combination of *FS* and *BE*, so it is slower than *FS* but faster than *BE*. But as *FSBE* considers both elimination and addition in each step, it usually produces better output than both *BE* and *FS*.

6. RELATED WORK

A large corpus of work has tackled the problem of ranking the results of a query. In the documents world, the most popular techniques are tf-idf based (Salton, 1989) ranking functions, like BM25 (Robertson & Walker, 1994), as well as link-structure-based techniques like PageRank (Brin & Page, 1998) if such links are present (e.g., the Web). In the database world, automatic ranking techniques for the results of structured queries have been recently proposed ([Agrawal, Chaudhuri, Das, & Gionis, 2003], [Chaudhuri, Das, Hristidis, & Weikum, 2004], [Su, Wang, Huang, & Lochovsky, 2006]). In addition to ranking the results of a query, there has been recent work (Das, Hristidis, Kapoor, & Sudarshan, 2006) on ordering the displayed attributes of query results.

Both of these tuple and the attribute ranking techniques are inapplicable to our problem. The former inputs a database and a query, and outputs a list of database tuples according to a ranking function, and the latter inputs the list of database results and selects a set of attributes that "explain" these results. In contrast, our problem inputs a database, a query log, and a new tuple, and computes a set of attributes that will rank the tuple high for the skylines of as many queries in the query log as possible.

Our work differs from the extensive body of work on *feature selection* (Guyon, & Elisseeff, 2003) because our goal is very specific – to enable a tuple to be highly visible to the users of the database as well as stand out in the crowd of existing products – and not to reduce the cost of building a mining model such as classification or clustering.

Kleinberg, Papadimitriou, & Raghavan (1998) present a set of microeconomic problems suitable for data mining techniques; however no specific solutions are presented. Their problem closer to our work is identifying the best parameters for a marketing strategy in order to maximize the attracted customers, given that the competitor independently also prepares a similar strategy. Our problem is different since we know the competition (other data items). Another area where boosting an item's rank has received attention is Web search, where the most popular techniques involve manipulating the link-structure of the Web to achieve higher visibility (Gori & Witten, 2005).

Computing frequent itemsets is a popular area of research in data mining and some of the best known algorithms include Apriori (Agrawal & Srikant, 1994) and FP-Tree (Han, Pei, & Yin, 2000). In frequent itemset mining, a subset of items are predicted which are frequent (occurs together more than a threshold) in the transaction database. Here, a frequent itemset is basically a subset of a transaction. Our problem is the opposite, we want to identify the

subset of attributes (items) which to retain for the new tuple *t* such that *t* becomes a superset of a skyline (transaction).

The works on dominant relationship analysis (Li, Ooi, Tung, & Wang, 2006) and dominating neighborhood profitably (Li, Tung, Jin, & Ester, 2007) are related to our work. The former tries to find out the dominant relationship between products and potential buyers where by analyzing such relationships, companies can position their products more effectively while remaining profitable, and the latter introduces skyline query types taking into account not only min/max attributes (e.g., price, weight) but also spatial attributes (e.g., location attributes) and the relationships between these different attribute types. Their work aims at helping manufacturers choose the right specs for a new product, whereas our work aims at choosing the attributes subset of an existing product for advertising purposes.

Skyline query processing has been well investigated recently. Several techniques have been proposed for efficient skyline query processing ([Borzsonyi, Kossmann, & Stocker, 2001], [Tan, Eng, & Ooi, 2001], [Kossmann, Ramsak, & Rost, 2002], [Papadias, Tao, Fu, & Seeger, 2003]). There has been recent work on categorical skylines (Sarkas, Das, Koudas, & Tung, 2008), where the authors proposed a method for maintaining efficiently the skylines of streaming data with partially ordered, categorical attributes. One main difference of our work with the existing works is that we consider Boolean skylines and our goal is not to propose a method to efficiently process or maintain the skylines, instead we use skylines as a query semantic where a new tuple can be visible for a maximum number of queries.

7. CONCLUSIONS

In this paper we consider a problem that has applications in marketing and product design - given a new product (a tuple), to select a small subset of attributes to highlight such that the product stands out in a crowd of existing competitive products and is widely visible to the pool of potential customers. A product is considered to be visible to a customer if it occurs in the skyline of the query posed by the customer. This problem variant poses new challenges that cannot be solved optimally using prior techniques, hence we develop novel optimal algorithm based on the signature tree data structure as well as approximate algorithms to solve the problem.

Clearly, the definition of visibility of a product to customers can be extended beyond the concept of skylines. As future work we are considering other interesting definitions of product visibility, and investigating whether signature trees and similar techniques can be used for solving such problems.

8. REFERENCES

Agrawal, S., Chaudhuri, S., Das, G., & Gionis, A. (2003). Automated Ranking of Database Query Results. *CIDR* 2003.

Agrawal, R., & Srikant, R. (1994). Fast Algorithms for Mining Association Rules. VLDB 1994: 487-499

Borzsonyi, S., Kossmann, D., & Stocker, K. (2001). The Skyline Operator. *ICDE* 2001.

Brin, S., & Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine. *WWW Conference*, 1998

Chaudhuri, S., Das, G., Hristidis, V., & Weikum, G. (2004). Probabilistic Ranking of Database Query Results. *VLDB* 2004

Chen, Y., & Chen, Y. (2006). On the Signature Tree Construction and Analysis. *IEEE Trans. Knowl. Data Eng.* 18(9): 1207-1224

Das, G., Hristidis, V., Kapoor, N., & Sudarshan, S. (2006). Ordering the Attributes of Query Results. *SIGMOD* 2006.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(mar):1157–1182

Gori, M., & Witten, I. (2005). The bubble of web visibility. *Commun. ACM* 48, 3 (Mar. 2005), 115-117

Han, J., Pei, J., & Yin, Y. (2000): Mining Frequent Patterns without Candidate Generation. *SIGMOD* 2000: 1-12.

Kleinberg, J., Papadimitriou, C., & Raghavan, P. (1998). A Microeconomic View of Data

Mining. *Data Min. Knowl. Discov*. 2, 4 (Dec. 1998), 311-324

Kossmann, D., Ramsak, F., & Rost, S. (2002). Shooting Stars in the Sky: an Online Algorithm for Skyline Queries. *VLDB* 2002.

Li, C., Tung, A. K. H., Jin, W., & Ester, M. (2007). On Dominating Your Neighborhood Profitably. VLDB 2007: 818-829

Li, C., Ooi, B. C., Tung, A. K. H., & Wang, S. (2006). DADA: a Data Cube for Dominant Relationship Analysis. SIGMOD Conference 2006: 659-670

Miah, M. (2009). An Optimal Signature-Tree based Algorithm for Selecting Attributes for Maximum Visibility. *International Conference on Information Technology (ICIT)* 2009.

Miah, M., Das, G., Hristidis, V., & Mannila, H. (2009). Determining Attributes to Maximize Visibility of Objects. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 2009, vol. 21 no. 7, pp. 959-973.

Papadias, D., Tao, Y., Fu, G., & Seeger, B. (2003). An Optimal and Progressive Algorithm for Skyline Queries. *ACM SIGMOD* 2003

Robertson, S. E., Walker, S. (1994). Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. *SIGIR* 1994

Salton, G. (1989). Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. *Addison Wesley*, 1989

Sarkas, N., Das, G., Koudas, N., & Tung, A. K. H. (2008). Categorical skylines for streaming data. *SIGMOD Conference* 2008: 239-250

Su, W., Wang, J., Huang, Q., & Lochovsky, F. (2006). Query Result Ranking over E-commerce Web Databases. *ACM CIKM* 2006

Tan, K., Eng, P., & Ooi, B. C. (2001): Efficient Progressive Skyline Computation. *VLDB* 2001.

Appendices

Appendix 1: Some Useful Definitions and Notations

***Database*:** Let $D = \{t_1 \ldots t_N\}$ be a collection of tuples with Boolean attributes over the attribute set $A = \{a_1 \ldots a_M\}$, where each tuple t is a set of attributes. Considering a car database, a t has the actual attribute names which represents that an attribute is present in the tuple (e.g., *AC*).

***Tuple Compression*:** Let t be a tuple and let t' be a subset of t with m attributes. Thus t' represents a compressed representation of t.

Query: We view each query as a subset of attributes where users search for a product specifying their attributes of interest.

***Query Log*:** Let $Q = \{q_1 \ldots q_R\}$ be collection of queries where each query q defines a subset of attributes.

***Skyline*:** Given a set of points, the skyline comprises the points that are not dominated by other points. A point dominates another point if it is as good or better in all dimensions and better in at least one dimension (Tan, Eng, & Ooi, 2001). Consider a common example in the literature, "*choosing a set of hotels that is closer to the beach and cheaper than any other hotel in distance and price attributes respectively from the database system of the travel agents'* (Kossmann & Stocker, 2001)". Figure 1 illustrates this case in *2*-dimensional space, where each point corresponds to a hotel record. The *x*-axis specifies the room price of a hotel, and the *y*-axis specifies its distance to the beach. Clearly, the most interesting hotels are the ones *{a*, *g*, *i*, *n}*, called *skyline*, for which there is not any other hotel in *{a*, *b*, . . . , *m*, *n}* that is better on both dimensions.

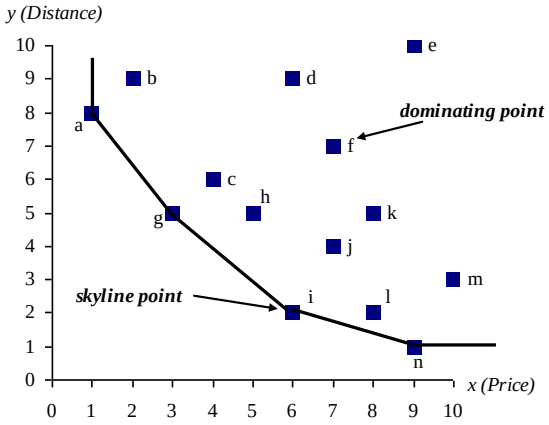


Figure 1. Skyline Example

***Skyline Log*:** Let $S = \{s_1 \ldots s_L\}$ be collection of skylines where each skyline s defines a subset (i.e., projection) of attributes for which any data point (tuple) remains on the skyline. For example, if a user poses a query q = "Select * from Cars where Make = Honda and AC = yes and Power Windows = yes", and the database has three cars t_1 = <*Toyota, AC, Power Windows*>, t_2 = <*Honda, AC, Power Brakes*> and t_3 = <*Nissan, AC, Power Brakes*>. We can see it easily from the skyline definition that the cars t_1 and t_2 will be on the skyline of q, which are not dominated by any other cars (t_3 here) present in the database based on the attributes asked by the query q. We do not store the actual skyline data points (all attributes present in the tuple) such as t_1 and t_3 in skyline log, instead the set of attributes for which a data point is visible on the skyline. Here, t_1 = <*Toyota, AC, Power Windows*> is visible on the skyline of q because it has attributes {*AC, Power Windows*} present asked by q. So,

the skyline we define here as s_1 = {*AC, Power Windows*}. Similarly skyline of for t_2 is s_2 = {*Honda, AC*} for which t_2 is on the skyline of q. Skylines log contains all such skylines for the query log.

Appendix 2: Skylines of queries

Skyline ID	*Query ID*	*Car ID (cars on the skyline)*	*Attributes for which the car is on the skyline*
s_1	q_1	t_4, t_5	{AC, Four Door}
s_2	q_2	t_3, t_4	{AC, Power Doors}
s_3	q_3	t_1, t_4, t_6	{Four Door, Power Doors}
s_4	q_4	t_3, t_4	{Power Brakes, Power Doors}
s_5	q_5	t_2, t_7	{Turbo}
s_6	q_5	t_3	{Auto Trans}

Table 1. Skylines of the Queries

Appendix 3: Optimal Naive Algorithm

The problem is proved to be NP-hard (Miah et al. 2009). A naïve optimal approach to find the subset of attributes (*m* attributes) to retain for the new tuple *t* to maximize the number of queries which will have *t* on the skylines as follows:

I. Generate all possible *m*-attributes candidate sets.
II. For each candidate set *c* in step (I), scan the skyline log and find for how many queries the skylines are the subsets of *c*.
III. Return the candidate set *c* with the highest count.

In practice, the Naïve algorithm is not feasible when the number of attributes is large since the algorithm has to generate a huge number of possible candidate sets. If the database has a total of *M* attributes and we want to retain *m* attributes, then there are total $\binom{M}{m}$ possible attribute sets which can be a very large number.

Appendix 4: Signature Tree

Signature Tree and its Construction: Once we have the skylines found then we can create signature for each skyline. We keep attributes sorted in each skyline. Creating signatures is as follows. We first initialize a bit vector of length *M* (total number of attributes in the database) with default value 0 for a skyline. In our running example, we have *M* = 6 attributes *(AC, Auto Trans, Four Door, Power Brakes, Power Doors, and Turbo)*, so the length of the signature for each skyline would be 6 and the initialize vector of the signature is 000000. Then for each attribute present in the skyline, we set that the corresponding bit in the bit vector to be 1. For example, for skyline s_1 = {*AC, Four Door*}, the signature is *101000*. A signature file contains the signatures of all the skylines (or transaction traditionally). Table 2 shows the signatures of the skylines (signature file) for our running example.

Skyline ID	*Query ID*	*Signature*
s_1	q_1	101000
s_2	q_2	100010
s_3	q_3	001010
s_4	q_4	000110
s_5	q_5	000001
s_6	q_5	010000

Table 2. Signature of Skylines (Signature File)

Definition (Signature tree): A signature tree for a signature file $S = s_1.s_2...s_n$, (where $s_i \neq s_j$ for $i \neq j$ and $|s_k| = d$ for $k = 1, \ldots, n$) is a binary tree T such that

i. For each internal node of T, the left edge below it is always labeled with 0 and the right edge is always labeled with 1.
ii. T has n leaves labeled 1,2, …,n, used as pointers to n different positions of s_1, s_2, …, and s_n in s. Let v be a leaf node. Denote $p(v)$ the pointer to the corresponding signature.
iii. Each internal node v is associated with a number, denoted by $s_k(v)$, denoting which digit will be checked.
iv. Let, $i_1,\ldots,i_h$ be the numbers associated with the nodes in a path from the root to a leaf v labeled i. Then, this leaf node is a pointer to the ith signature in S, i.e., $p(v) = i$. Let $p_1,\ldots,p_h$ be the sequence of labels of edges on this path. Then, $(j_1, p_1)\ldots(j_h, p_h)$ makes up a signature identifier for s_i, $s_i(j_1, \ldots, j_h)$.

Creating Balanced Signature Tree: A balanced signature tree is a signature tree which is completely or almost evenly balanced. The method of building a balanced signature tree is described below. The tree might not be always perfectly balanced, but it would be close to being evenly balanced.

A signature file $S = s_1.s_2 \ . \ . \ . \ s_n$ can be considered as a Boolean matrix. We use $S[i]$ to represent the ith column of S. For our example above, we have the digits of signature represented for the attributes as follows:

Attribute	AC	Auto Trans	Four Door	Power Brakes	Power Doors	Turbo
Digit	1	2	3	4	5	6

We calculate the weight of each $S[i]$, i.e., the number of 1's appearing in $S[i]$, denoted $w(S[i])$. Then, we choose a j such that $|w(S[i]) - n/2|$ is minimum. Here, the tie is resolved arbitrarily. Using this j, we divide S into two groups $g_1 = \{s_{i1}, s_{i2}, \ . \ . \ . \ , s_{ik}\}$ with each $s_{ip}[j] = 0$ ($p = 1, \ . \ . \ . \ , k$) and $g2 = \{s_{ik+1}; s_{ik+2} ; \ . \ . \ . \ ; s_{iN}\}$ with each $s_{iq}[j] = 1$ ($q = k + 1, \ . \ . \ . \ , n$); and generate a tree as shown in Figure 4(a). In fact, we partition the signatures based on the value on column j; signatures with value 0 on column j go into one group and signatures with value 1 on column j go into another group. In a next step, we consider each g_i ($i = 1, 2$) as a single signature file and perform the same operations as above, leading to two trees generated for g_1 and g_2, respectively. Replacing g_1 and g_2 with the corresponding trees, we get another tree as shown in Figure 4(b). We repeat this process until the leaf nodes of a generated tree cannot be divided any more. Considering our running example, we can see that at the first time the sum of 1's in each column $w(S[i])$ is as follows: column 1 (AC) = 2, column 2 = 1, column 3 = 2, column 4 = 1, column 5 = 3, and column 6 = 1. Here, $n = 6$ which is the total number of skylines. So, column 5 has the minimum value for $|w(S[i]) - n/2|$ which is $(3 - 6/2) = 0$. So we choose column 5 which is *Power Doors* as the root of the tree. We follow the same process for each sub-tree from the root. In Figure 4(a), $g_1 = \{s_1, s_5, s_6\}$ and $g_2 = \{s_2, s_3; s_4\}$; and, in Fig. 6(b), $g_{11} = \{s_5, s_6\}$, $g_{12} = \{s_1\}$, $g_{21} = \{s_3, s_4\}$, and $g_{22} = \{s_2\}$. Figure 5 shows the complete signature tree built for the skylines of our running example.

At the leaf node of the tree, we keep information for the skyline as well as the query where it came from. As we recall, our goal is to maximize the number of queries for which the new tuple will be visible on the skylines. We do not want to count skylines from the same query for a candidate set more than once.

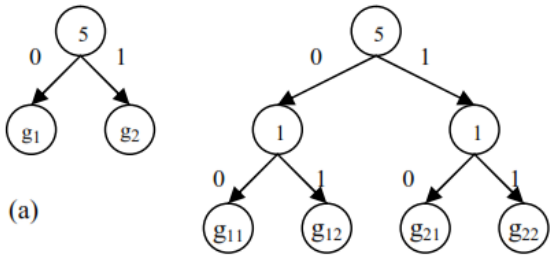


Figure 4. Process of Building Signature Tree

At this step we generate the signature tree only for the skylines with less than or equal to *m* attributes. The reason we ignore the skylines with more than *m* attributes is that none of them can eventually be subset of any *m*-attributes candidate set which we generate in next step. This will be an efficient technique where there are many skylines which have more than *m* attributes present in the skyline log.

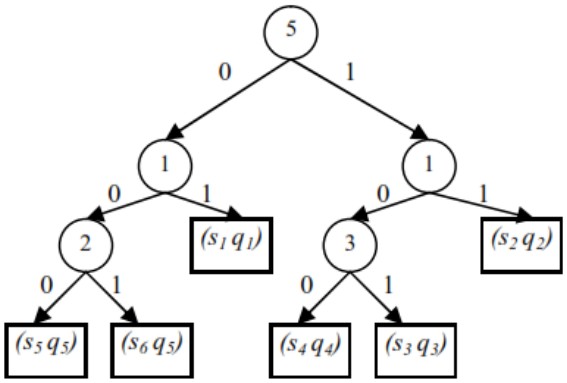


Figure 5. Signature Tree for the Skylines

Appendix 5: Traditional Approach of Searching the Signature Tree

We search the signature tree for the new tuple (*m* attribute set). As in the *Apriori* algorithm (Agrawal & Srikant, 1994), we start with frequent *1*-itemsets (attribute sets). A *minimum support* is used such that when we select top-*m* attributes for the new tuple *t*, then *t* should be on the skylines for the number of queries at least or equal to the *minimum support*. A minimum support is the lower bound such that at least these many queries should have the new tuple visible on their skylines. We use a heuristic method to select a good *minimum support*. We first use a fixed value, for example 1% and execute the algorithm. Then we change the minimum support as required, for example if we find no queries for the new tuple then we decrease the minimum support and if too many queries are found then we increase the minimum support until a good value for minimum support is set. Using the minimum support we generate frequent *1*-itemsets (attribute sets) from the skylines. Here we only consider the attributes which are present in the new tuple to be advertised. One approach now could be to generate all possible *m*-attribute sets using *apriori-gen* function in Figure 3, and then search the tree. We can search the signature tree as follows:

i. Create signature for each of the *m*-attribute candidate sets. Let s_t be the candidate set signature. The *i*th position of s_t is denoted as $s_t[i]$. During the traversal of a signature tree, the inexact matching is done as follows:

   a. Let *v* be the node encountered and $s_t[i]$ be the position to be checked.
   b. If $s_t[i] = 0$, we move to the left child of *v*.
   c. If $s_t[i] = 1$, both the right and left child of *v* will be explored.

   In fact, this process just corresponds to the signature matching criterion, i.e., for a bit position *i* in s_t, if it is set to 0, the corresponding bit position in *s* must be set to 0; if it is set to 1, the corresponding bit position in *s* can be 1 or 0. In a traditional signature tree, a query *q* is passed to the tree and finds the transactions (leaf nodes of the tree) which are possibly the supersets of *q* (i.e., how many transactions will be retrieved by the query). But our problem is different. We pass a candidate *m*-attribute set *c* to the tree and find the skylines (leaf nodes) which are possibly subsets of *c*. Searching of the tree is done in a depth-first manner. When we reach a leaf node, we match all the signatures of the leaf node with *m*-attribute candidate set *c*. Here skylines have to be subsets of *c*. We keep a count for each candidate set *c* that how many skylines have been found as subsets of *c*. Here, we count each query only once. For example, considering our running example, if we find two skylines s_5 and s_6 are the subsets of any candidate set c_i, we only count them once because both come from the same query q_5. As we recall, our problem is not to maximize the number of skylines, but to maximize the number of queries which will have the new tuple on their skylines. We remove the candidate set *c* if the total count for it is less than the *minimum support*.

ii. For all m-attributes candidate sets found in step (i), we simply return the set that has the highest count.

Appendix 6: Major Problems of Traditional Searching of the Signature Tree

There are two major problems with the traditional approach of searching the trees: (a) the number of candidate sets can be huge as there is no pruning at intermediate steps by searching the tree, and (b) small itemsets would get an unfairly small count because it increases the count of a candidate if it satisfies whole skyline itemsets in the signature tree. Hence, in order to be able to grow the candidate itemsets and not start directly from *m*-itemsets, we start generating and searching the tree in order to increase the count of a candidate *k*-itemset for every query it has a chance to cover if (*m*-*k*) items are added. For instance, the *2*-itemset 110000 has a chance to cover 110100 if 1 more item is added. So we follow a new method where for each *k*-itemset we navigate the signature tree from top to bottom and only prune subtrees that need more than (*m*-*k*) additional items to be covered.

Appendix 7: System Configuration and Datasets used for the Experiments

System Configuration: We used Microsoft SQL Server 2000 RDBMS on a P4 3.2-GHZ PC with 1 GB of RAM and 100 GB HDD for our experiments. We implemented all algorithms in C#, and connected to the RDBMS through ADO.

Dataset: We use an *online used-cars dataset* consisting of 15,211 cars for sale in the Dallas area extracted from autos.yahoo.com. There are 32 Boolean attributes such as *AC, Power Locks*, etc. We used a real query log of 185 queries created by university users, as well as synthetic query logs of 1000, and 100000 queries. In the synthetic query logs, each query specifies 1 to 5 attributes chosen randomly distributed as follows: 1 attribute – 20%, 2 attributes – 30%, 3 attributes – 30%, 4 attributes – 10%, 5 attributes – 10%. That is, we assume that most of the users specify two or three attributes.

Applying Business Intelligence Concepts to Medicaid Claim Fraud Detection

Leanndra Copeland
l-copeland@nvdetr.org
Nevada Department of Employment, Training and Rehabilitation
Carson City, NV 89713, USA

Dana Edberg
dte@unr.edu
Department of Information Systems

Anna K. Panorska
ania@unr.edu
Department of Mathematics and Statistics

Jeanne Wendel
wendel@unr.edu
Department of Economics

University of Nevada, Reno
Reno, NV 89557, USA

Abstract

U.S. governmental agencies are striving to do more with less. Controlling the costs of delivering healthcare services such as Medicaid is especially critical at a time of increasing program enrollment and decreasing state budgets. Fraud is estimated to steal up to ten percent of the taxpayer dollars used to fund governmentally supported healthcare, making it critical for government authorities to find cost effective methods to detect fraudulent transactions. This paper explores the use of a business intelligence system relying on statistical methods to detect fraud in one state's existing Medicaid claim payment data. This study shows that Medicaid claim transactions that have been collected for payment purposes can be reformatted and analyzed to detect fraud and provide input for decision makers charged with making the best use of available funding. The results illustrate the efficacy of using unsupervised statistical methods to detect fraud in healthcare-related data.

Keywords: business intelligence, government information systems, healthcare information technology, fraud, statistical analysis, unsupervised methods

1. INTRODUCTION

Publicly funded agencies in the U.S. face taxpayer demand to extend and improve services while also progressively lowering costs. These pressures are especially significant within the area of governmentally supported healthcare (Ryan, 2011).

The Centers for Medicare and Medicaid Services (CMS) estimate total U.S. health care spending in 2009 reached $2.5 trillion or 17.6% of gross domestic product (CMS, 2010). While it is difficult to pinpoint the exact amount of fraud in healthcare transactions, federal agencies estimate that from 3% to 10% of expenses are fraudulent, with 10% being the most accepted figure (Heaphy, 2011). Even using the most conservative estimate, it means that over $75 billion per year targeted for providing healthcare is stolen from taxpayer funds through fraudulent activities. To help combat this problem, the federal government has established Medicaid Fraud Control Units and State Program Integrity Units providing assistance and oversight for healthcare payment processes (CMS, 2011), but there is so far no evidence that these mechanisms have yielded significant improvement in fraud detection and protection.

Commercial enterprises frequently use business intelligence (BI) systems to help identify and control fraud (Han & Kamber, 2006; Kotsiantis, Koumanakos, Tzelepis, & Tampakas, 2006; Wegener & Rüping, 2010). BI systems consist of methods of gathering data, data storage (typically termed a "data warehouse"), and analytical tools such as visualization programs, statistical methods and data mining algorithms (Negash, 2004). The information gleaned from BI is used to provide decision makers with accurate, timely, well-presented information. BI systems are a platform for organizing and analyzing data from disparate sources to provide meaningful information for decision makers (Davenport & Harris, 2007; Negash, 2004).

Health insurance organizations have applied this same technology successfully to monitor fraudulent activities (Sokol, Garcia, Rodriguez, West, & Johnson, 2001; Wang & Yang, 2009; Yang & Hwang, 2006). While authors emphasize the importance of using BI to support governmental decision making (Davenport & Jarvenpaa, 2008), actual implementation of these systems is problematic. Some government agencies experience challenges implementing BI due to the short-term funding cycles required by governing bodies, a lack of personnel with knowledge of BI, restricted funding, concerns about privacy, incomplete data, and poor integration of available data (Harper, 2004; Rosacker & Olson, 2008; Vann, 2004; Wilkin & Riddett, 2009).

A recent report from the U.S. Government Accountability Office (GAO, 2011) recommended that states make more effective use of information technology to help detect healthcare-related fraud, but also highlighted the difficulties experienced in creating and accessing a nationwide data repository for this function. The report emphasized the challenges in creating a fully integrated dataset to support inquiries from state agencies (GAO, 2011).

This paper addresses the use of BI technology for detecting fraud in a state's Medicaid payment system. We discuss how a state could use its existing data to create a data warehouse, and then illustrate how statistical methods could be applied to detect fraudulent Medicaid claims. Given the rising costs of healthcare, and the shrinking operating budgets of state government, it is of critical importance that fraud control units incorporate BI technology for effective fraud detection.

The next section of this paper presents existing research on methods of detecting fraud, highlighting studies aimed at identifying fraud in healthcare. The third section describes our study performed in one state using Medicaid claim data, and the fourth section briefly discusses the practical considerations of implementing BI to help detect Medicaid fraud.

2. METHODS OF DETECTING FRAUD

There are two main strategies for detecting fraud: auditing and statistics. Auditing strategies require the use of trained personnel to evaluate the process and/or product, while statistical methods rely on large data sets to identify potential anomalies. A summary of the strategies for detecting fraud is provided in Table A-1 (see Appendix), and each is described in the following sub-sections.

Auditing Strategies

When auditing healthcare systems, medical and claims experts are hired to review transactional claims on a case-by-case basis to identify

anomalies based on the knowledge of those reviewing the claims (Yang & Hwang, 2006). Auditing strategies frequently use random stratification sampling methods to obtain samples from a spectrum of different claim types (Buddhakulsomsiri & Parthanadee, 2008), but cannot pinpoint all suspicious claims among millions of claims in a data set.

In a study of healthcare-related fraud, claims for durable medical equipment (DME) from two multi-county areas within a region served by an insurance carrier were analyzed (Wickizer, 1995). Part of the study utilized an audit strategy where nurse analysts reviewed DME claims to verify accuracy. Four different types of DME were examined. As testament to the time factor in using audits, this study used a sample size of just 231 observations. The researcher examined twenty-one months (January 1990 – September 1991) of claims data in which four variables provided measurement for DME utilization: (1) number of order requests per month, (2) submitted charges per month, (3) Medicare-allowed payments per month, and (4) percentage of DME requests denied per month. Data on other covariates were gathered to control for external factors that could influence DME utilization, such as the number of hospital discharges per 1,000 Medicare beneficiaries. The findings showed that DME utilization management programs reduced the number of requests, submitted charges, and Medicare payments in three out of the four targeted DME items.

While auditing strategies tend to be accurate in finding fraud, they are costly and time-consuming to perform on the large number of transactions processed in the healthcare industry (Yang & Hwang, 2006). Thus, these strategies may not be feasible for detecting fraud in government organizations that are trying to make the best use of limited resources.

Statistical Strategies

Statistical fraud detection strategies rely on analytical methods such as correlation and regression to evaluate large data sets (Bolton & Hand, 2002). Some studies have pointed out that finding the source of fraud (insured, provider, etc) using statistical methods is far more efficient than analyzing individual claims (Ortega, Figueroa, & Ruz, 2006; Yang & Hwang, 2006).

Statistical strategies are classified as supervised or unsupervised methods (Bolton & Hand, 2002). Supervised methods require samples from both known fraudulent and non-fraudulent records in order to model the distinct characteristics of each. The data is labeled by human experts prior to processing through sophisticated computer data mining algorithms. Unsupervised methods, on the other hand, do not require any prior knowledge of the relative legitimacy of the data and the data is unlabeled. These two methods could be considered endpoints on a continuum of statistical strategies, with hybrid or semi-supervised methods sitting in the middle (Laleh & Abdollahi Azgomi, 2009). Semi-supervised methods use some data that is labeled, and some unlabeled data that is evaluated during program processing. The labeled data must be identified prior to input and requires pre-knowledge of fraudulent transactions for modeling purposes. To simplify the discussion of statistical methods, the next two sub-sections discuss studies that focus on the two endpoints on the methods continuum.

Supervised Statistical Methods

A key issue in the use of supervised statistical methods is the need to identify fraudulent claims prior to using the data for further processing. An example of this constraint is a study using a multi-layer perceptron network to classify general practitioner (GP) physician profiles into categories ranging from normal to abnormal (He, Wang, Graco, & Hawkins, 1997). The study required an auditing portion to develop the supervised methods. Physicians, hired as expert consultants, identified 28 features which summarized a GP's practice over a year. The classified sample was used to train an automated classification system. The sample consisted of 1,500 randomly selected GP profiles from Australian physicians who participated in Medicare. The physicians serving as consultants classified all 1,500 profiles based on 28 distinct features before the sample was divided into two groups with 750 profiles for the training set and 750 profiles for the test set. The researchers concluded that a two-class neural network classification system was a viable method for detecting fraud. A problem with the method employed in the study is that it is not easily replicable in a governmental organization because of the expense involved with hiring medical experts to review such a large number of claims and create valid feature variables. In

addition, the necessary software algorithms require personnel skilled in statistical programming and data mining operations.
Another study required meetings with medical experts to assist in developing a set of variables used to discriminate between fraudulent and honest claims (Ortega, et al., 2006). This study classified 125 distinct features to four different areas/parties where fraud can occur: medical claims, the insured, the medical professional, and the employer. In addition, this study incorporated feedback from each model input in the other three sub-models. As is common in studies detecting fraud, a full discussion of the results was prohibited due to a disclosure agreement between the authors and the insurance company that provided the data. However, it was proclaimed that the model accelerated detection on average 6.6 months earlier than standard audit strategies.

Another issue in the use of supervised statistical methods is the relative balance between legitimate and fraudulent transactions. Legitimate transactions far outweigh fraudulent ones in any empirical dataset. Creating models from these unbalanced classes can cause misspecification (Bolton & Hand, 2002). Finally, the most critical issue is that supervised models cannot detect new types of fraud because the models are created from past fraud strategies (Bolton & Hand, 2002; Laleh & Abdollahi Azgomi, 2009). Despite these issues, supervised methods are widely used for fraud detection in healthcare and are supported by technologies such as neural networks , decision trees, fuzzy logic, and Bayesian networks (Li, Huang, Jin, & Shi, 2008).

Unsupervised Statistical Methods

Unsupervised statistical methods determine and tag outliers in a data set so that those outliers can then be marked for potential investigation. Unsupervised methods first use technology to identify potentially fraudulent transactions, and afterwards require the use of expertise to determine the legitimacy of those transactions. The assumption is that fewer transactions will have to be investigated than in supervised methods, and the investigation can be performed by less costly personnel (Laleh & Abdollahi Azgomi, 2009). Unsupervised methods use clustering as a popular tool for detecting anomalous data (Bolton & Hand, 2002).

Using an unsupervised method, a study reviewed the medical insurance claims of 22,000 providers to test an electronic fraud detection (EFD) program (Major & Riedinger, 2002). The technique compared individual provider characteristics to their peers. The researchers recommended that provider comparisons be grouped according to similar characteristics, such as the same organizational structure, specialty, and geographic location. The EFD developers examined 27 behavioral heuristics in five categories: financial (the flow of dollars), medical logic (whether a medical situation would normally happen), abuse (frequency of treatments), logistics (place, time and sequence of activities), and identification (how providers present themselves to the insurer). Validation of the model was yet to be determined at the time of publication, but the model did alert officials to over 800 suspicious providers and resulted in the launch of 23 investigations (Major & Riedinger, 2002).

Another approach to unsupervised fraud detection is a probabilistic model called Benford's Law. Benford's Law, sometimes referred to as the first-digit law, states that the first significant digit of many data sets follows a known frequency pattern (Nigrini & Mittermaier, 1997). Benford's Law has been applied to fraud detection (for tax data) through the development of the Distortion Factor (DF) Model (Nigrini, 1992; Watrin, Struffert, & Ullmann, 2008). The DF model compares the first digit frequencies of observations in a data set to the expected frequencies of Benford's Law. The first digit distribution of many data sets follows Benford's Law, such as the one-day returns on the Dow Jones Industrial Average and the Standards and Poor's Index, street address, and many others data sets.

Unsupervised statistical methods may be more cost effective for government agencies than auditing or supervised statistical methods. Unsupervised methods use standard statistical processing, so it may be easier for government agencies to find appropriate personnel as compared to the medical knowledge required for supervised methods. In addition, the initial detection of potential fraud is performed by technology, allowing for greater focus of expert time on those transactions that have a greater probability of fraud. An advantage of unsupervised methods is that they can detect new types of fraud. On the other hand, unsupervised methods require expertise in the

initial development of potential factors that should be analyzed for outliers as well as the creation of an appropriate dataset for processing. In addition, the efficacy of unsupervised statistical methods is relatively untested in the literature (Bolton & Hand, 2002).

Purpose of this Study

This study contributes to the literature by exploring the use of unsupervised statistical methods for detecting healthcare fraud. The use of electronic healthcare claims lends itself to evaluation through BI tools such as statistically-based analytical methods. The goal of this study is to use statistical methods to improve the accuracy of detecting fraud, while minimizing the overall cost of system implementation for a government agency. This study was motivated by a practical need to create a simple and cost effective method of analyzing existing claims data to identify potential fraud. We demonstrate how that data might be reformatted and used to provide additional information for decision makers who are attempting to detect and control fraud.

The next section of this paper explores the use of unsupervised statistical methods to detect fraud in Medicaid claims for the state of Nevada.

3. APPLYING UNSUPERVISED STATISTICAL METHODS TO MEDICAID CLAIMS

Nevada is the seventh-largest state geographically, but with a relatively small population of 2.7 million people. About 10% of the Nevada population was enrolled in Medicaid in 2009 (as compared to an average 19% enrollment rate nationwide) and total Medicaid expenses for 2009 in Nevada were approximately $1.3 billion (Kaiser, 2009).

A recent spree in durable medical equipment, prosthetic, orthotic devices, and/or disposable medical supplies (DMEPOS) fraud in the state of Nevada prompted state authorities to explore whether BI might help the state become more effective at detecting fraud. DMEPOS is defined as equipment that is appropriate for in-home use and benefits the patient medically. DMEPOS may consist of items that can be used a repeated number of times or may be disposable supplies which are not reusable (NVHHS, 2009). State authorities identified a particular DMEPOS item, disposable diapers, as being most appropriate for initial exploration. Diaper fraud is attractive to fraudsters because it is a high-volume item requiring relatively little medical expertise to process. Over the five year time frame of data used for this study, Nevada reimbursed 321 supplier companies for briefs, diapers and pads.

Data Used for Evaluation

The Nevada Department of Health and Human Services provided de-identified Medicaid claims data that linked provider, facility, and prescription claim transaction records over a five-year time period from January 2005 to December 2009. During this time, Medicaid reimbursed 693 DMEPOS supply companies for a total of $87,340,766. Data came from three different payer organizations, and was presented to us using three different formats. The data was delivered in comma delimited ASCII files.

Database Design, Extract, Transform and Load

Some researchers estimate that data preparation consumes 80% of the time in a fraud detection project (Li, et al., 2008; Lin & Haug, 2006; Sokol, et al., 2001). Database structures of raw claims data and electronic health records are designed to support financial transactions and health care delivery, rather than fraud detection or query development, and thus must be reshaped to support data analysis operations. We created a data warehouse from the data files that could be used to support multiple inquiries. Data preparation for this project was time-consuming. We estimate that data preparation took about 85% of overall project time. However, once the data was loaded in a data warehouse, it could be accessed in a variety of ways for different analytical applications so we anticipate that future data preparation time will be significantly less than the original development.

A normalized database design was created to store de-identified data about patients, providers and claims. The data warehouse used for this study was used for additional studies, so it was critical to create an adaptable and flexible design. Claims were subdivided into provider, facility and pharmacy categories to facilitate faster data access. The data warehouse was implemented using Microsoft SQL Server 2008R2. Data from the three different payer organizations was extracted, transformed and loaded (ETL) using both SQL Server Integration

Services and customized load routines. Since data formats differed among the three input sources, data had to be made consistent during the ETL process. The database contained a total of approximately 46.7 million claim records for the five year period. The data of interest for this study characterizes the DME suppliers enrolled to provide services to Medicaid patients and their claims. This subset consisted of about 10 million claim records.

Data Analysis

After the data was loaded in the data warehouse, data analysis proceeded iteratively to identify appropriate features and evaluate the data.

The claim records used included detailed information such as diagnosis codes, procedure (DMEPOS) codes, de-identified patient number, total charges claimed, etc. Every provider with a disproportionately high or low outcome for a given variable was assigned weighted points based upon total number of patients, total amount claimed, or length of company operation to ensure a variable did not disproportionately represent any provider with certain characteristics. A variable that resulted in high quality data was weighted more than variables with lower discriminatory power. Furthermore, the size of the company could have affected the outcome of a variable and was taken into consideration before assigning points. The assigning of weights will be addressed further in the results section.

Features that might help detect fraudulent activities were derived from the literature and from discussions with Nevada state authorities. A profile consisting of 12 features was ultimately created for each of the 321 DME suppliers providing incontinence briefs, diapers, or pads. A hindrance in fraud control efforts is the expurgation of public discourse about new fraud detection techniques to prevent alerting fraudsters. If criminals gain knowledge of how detection systems work, this could occlude the efficacy of new ideas before opportunity to detect fraud arises. Thus, academic literature rarely reveals the features used to isolate fraud (Bolton & Hand, 2002). The features created for this study were largely original and cannot be revealed due to an agreement with authorities from Nevada. Besides the censoring of enforcement techniques, provision of data sets and complete discussion of fraud study results are a rarity in academic literature (Bolton and Hand, 2002).

After the 12 features were solidified, analysis proceeded in three steps. First, the DMEPOS supplier's behavior was measured for each feature. Second, suppliers were compared against each other. If a supplier fell into the outlier range as determined by the upper or lower fifth percentile of any of the features, the supplier was assigned weighted points given the strength of the variable and the size of the supplier's transactions, as mentioned previously. Because the upper or lower fifth percentile cutoffs are assigned based on statistics, not logic, thought should be given to whether the statistical cutoff divides the groups into questionable and likely benign categories. Thus, the third step assists in this task by providing visualization through tables and graphs. Visualization techniques help show whether or not the feature variable divides the suppliers into useful categories with noticeable extremes. If not, the weight for the feature variable is lowered. The more points a supplier has after all twelve feature variables are analyzed, the more suspicious that supplier looks. The next subsection provides more detail about two of the twelve feature variables used in the study.

Results: Diapers per claim

Medicaid rules limit patients to 300 diapers per month. The more diapers supplied per claim, the more money a fraudulent company can make. If a supplier consistently orders 300 diapers per claim for multiple patients, this means that most of their patients need approximately ten diapers a day. This equates to changing a brief nearly every two and half hours around the clock. Patients may initially require more briefs because they need to stock up around the house and other frequented locations. For example, parents with newborn children have diapers in the car, living room, bedroom, etc. Adults needing briefs would go through the same transition and would require more in the beginning.

The results showed that suppliers whose average was over 272.5 diapers per claim fell into the 95 percentile. Using this metric, 16 companies were flagged as shown in Table 1.

Table 1. Suppliers Flagged for Feature 1

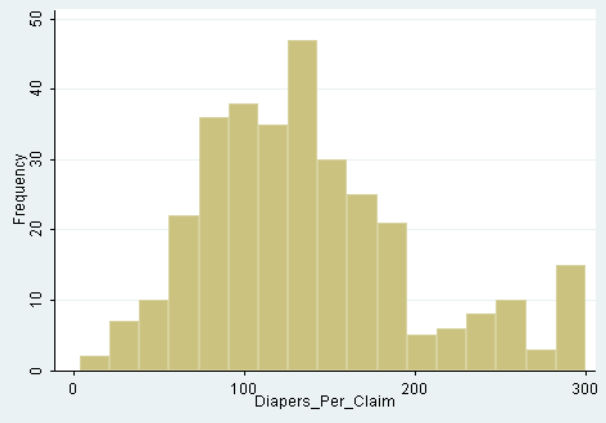
Supplier ID	Diapers Per Claim	Number of Unique Patients	Total Points
177	300	1	1 x 3 = 3
59	300	1	1 x 3 = 3
241	300	7	1 x 3 =3
307	300	45	3 x 3 = 9
179	300	152	5 x 3 = 15
100	300	138	5 x 3 =15
88	298	40	3 x 3 = 9
156	297	1	1 x 3 = 3
230	297	2	1 x 3 = 3
192	296	8	1 x 3 = 3
15	295	2	1 x 3 = 3
55	294	2	1 x 3 = 3
302	289	3	1 x 3 = 3
170	288	1	1 x 3 = 3
17	288	1	1 x 3 = 3
43	274	299	5 x 3 = 15

Before assigning points to the isolated companies, two things were considered: (1) some companies averaged a high number of diapers per claim, but only delivered to a few patients; and (2) the strongest variables separated suppliers into well-defined categories with a definite right tail.

Patients that need incontinence supplies have varying levels of bladder control; not all will need the maximum 300 briefs per month. To emphasize companies which consistently supplied a high average to numerous patients, a weighting system was implemented. The number of patients served determined the weight. Companies that supplied more than 50 patients were assigned five points. Companies with 20-49 patients were assigned three points, and companies with less than 20 patients were assigned one point.

Figure 1 illustrates the effectiveness of the feature variable at categorizing the suppliers. Figure 1 also demonstrates the power of visualization methods in BI, allowing a person to quickly see the anomalous suppliers. The histogram reveals that supplier behavior is skewed to the right showing a distinct right tail. Because there is a distinct distribution, the variable successfully identifies a marked right tail; therefore, more emphasis should be placed on suppliers isolated with this variable. Variables that divide suppliers into many categories were weighted 3 points. Revisit Table 1 to see the suppliers isolated by the diapers per claim feature variable.

Figure 13. Visualization Histogram for Diapers per Claim



The "diapers per claim" variable illustrates how an effective feature variable categorizes providers into a distribution with definite tails. Not all variables did such an effective job. The next section presents another feature that was less effective.

Results: Pre-authorization Requests

The number of pre-authorizations requested is an example of a variable with limited discriminative power.

If a supplier had no pre-authorization requests yet served many patients for an extended period, it may be considered suspicious. Fictitious organizations may prefer to limit their exposure to the system, whereas legitimate companies may require pre-authorizations at some point. Of the 321 DME companies that supplied briefs, 19 obtained pre-authorization for specialized orders.

Because only 6% of the suppliers needed a pre-authorization within the five year time frame, this feature variable did a poor job at categorizing the supplier companies. It essentially breaks the suppliers into two categories with the vast majority never utilizing the preauthorization system. Any supplier that

had at least one pre-authorization fell into the upper 95th percentile and was considered benign for this feature. This variable was not given much weight due to its inability to categorize the DMEPOS supply companies into many categories where a distinct tail can be seen; therefore, the variable is assigned a weight of one.

Next, the number of claims a supplier submitted is considered to further distribute points appropriately. Suppliers that submitted many claims were given more points. Suppliers that submitted less than 500 claims earned one point. Suppliers that submitted between 500 and 999 claims earned two points. Suppliers that submitted over 1,000 claims earned three points.

Table 2 details the results. "Claims" was chosen as the weight in acknowledgement of the need to stack up on supplies after initial diagnosis.

Table 2. Supplier Points Based on Preauthorization

Number of companies	Range of Count of Claims	Range of Count of Patients	Total points received per company
288	1-477	1-82	1
7	548-826	65-152	2
6	1151-4145	79-414	3

Overall Results

Table A-2 shows the suppliers that were flagged the most by the unsupervised statistical methods used for this study (see Appendix). The total amount of money spent on suspicious claims detected by this method totals $449,100, or 5.9 percent of the total amount spent on incontinence briefs during this five year period.

After presenting the results to the state fraud surveillance unit, it was determined that three of the six suppliers flagged were potentially fraudulent. Therefore, this method was believed by state authorities to have demonstrated its effectiveness in isolating suspicious suppliers.

Limitations

This is an exploratory study to help Nevada state authorities determine the applicability and effectiveness of BI for Nevada's Medicaid fraud detection. The results may be applicable only for this single state and may not be generalizable to the nationwide Medicaid claim population. Nevada's Medicaid population is significantly smaller than the national average and tends to contain more transient participants (Kaiser, 2009). There is little data available about the suppliers for Medicaid in the U.S., so we were not able to evaluate the comparability of Medicaid suppliers in Nevada to the rest of the U.S.

The point system applied in this study was used to explore the potential for relative weights in unsupervised methods. The weight system would need further evaluation to determine its most appropriate use.

4. CONCLUSIONS

There are three considerations if a governmental agency wished to implement unsupervised statistical methods for fraud detection. First, due to the dynamic nature of fraudulent activity, the way in which criminals commit fraud will evolve, as must the way in which the state goes about detecting it. The model presented here should be refined over time by dropping or adding relevant feature variables to continue being effective.

Second, a concern raised by state authorities about this procedure was that it only indentified companies that were no longer active; however, this method can easily be applied to real-time data to catch criminals before they go out of business. Real-time monitoring of provider behavior is a critical component of any medical fraud detection tool. This paper illustrates an effective method that could be incorporated with real-time claims data to achieve real-time business intelligence. The method presented is an analytic-based fraud detection tool that scores companies and isolates atypical providers.

Third, implementation of this model required knowledge of both standard statistical analysis and BI-related technology, as well as limited knowledge about the Medicaid application domain. Creation of the data warehouse required expertise in database design and ETL, while data access and analysis required skills in SQL programming and statistics. In order to determine the most appropriate feature variables, it was necessary to understand existing literature in healthcare fraud and to

gather information from state experts in Medicaid claims.

Fraud is a perpetually changing enterprise. Once the state detects a scheme, it should implement detection tools that use supervised methods to rapidly spot future schemes with similar characteristics. This detection method pushes criminals to constantly find new ways to steal money. The unsupervised statistical method presented in this paper should be used to continue scanning the data for new anomalies.

In these difficult financial times of shrinking state budgets and rising health-care costs, states need to target claims with a high probability of fraud so they can concentrate on stemming financial losses coming out of the taxpayers' wallets. Without implementing BI, the state will inevitably spend too much time reviewing honest claims.

This practical application of BI provides the opportunity for a government agency to reduce manpower and improve operational efficiency concurrently. The BI based analytic method explored in this study combines statistical methods with a data warehouse to turn data that is already available from claims processing into a new and powerful tool for detecting fraud.

5. REFERENCES

Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science, 17*(3), 235-249.

Buddhakulsomsiri, J., & Parthanadee, P. (2008). Stratified random sampling for estimating billing accuracy in health care systems. *Health Care Management Science, 11*(1), 41-54.

CMS. (2010). *National Health Expenditures 2009 Highlights*. Washington, DC: U.S. Department of Health & Human Services Retrieved from http://www.cms.gov/NationalHealthExpendData/downloads/highlights.pdf.

CMS. (2011). Medicaid Guidance Fraud Prevention. *State Program Integrity Support & Assistance* Retrieved July 14, 2011, from http://www.cms.gov/FraudAbuseforProfs/02_MedicaidGuidance.asp

Davenport, T. H., & Harris, J. G. (2007). *Competing on Analytics: The New Science of Winning*: Harvard Business Press.

Davenport, T. H., & Jarvenpaa, S. L. (2008). *Strategic Use of Analytics in Government*: IBM Center for the Business of Government.

GAO. (2011). *Fraud Detection Systems: Centers for Medicare and Medicaid Services Needs to Ensure More Widespread Use*. (GAO-11-475). Washington, DC: Report to Congressional Requesters Retrieved from http://www.gao.gov/new.items/d11475.pdf.

Han, J., & Kamber, M. (2006). *Data mining: concepts and techniques*: Morgan Kaufmann.

Harper, F. M. (2004). Data Warehousing and the Organization of Governmental Databases. In A. Pavlichev & G. D. Garson (Eds.), *Digital Government: Principles and Best Practices*. Hershey, PA: Idea Group, Inc.

He, H., Wang, J., Graco, W., & Hawkins, S. (1997). Application of neural networks to detection of medical fraud. *Expert Systems with Applications, 13*(4), 329-336.

Heaphy, T. J. (2011). Health Care Fraud Retrieved July 14, 2011, from http://www.justice.gov/usao/vaw/health_care_fraud/

Kaiser. (2009). State Health Facts Nevada: Total Medicaid Spending, FY 2009. *Individual State Profiles* Retrieved July 14, 2011, from http://www.statehealthfacts.org/profileind.jsp?cmprgn=1&cat=4&rgn=30&ind=177&sub=47

Kotsiantis, S., Koumanakos, E., Tzelepis, D., & Tampakas, V. (2006). Forecasting fraudulent financial statements using data mining. *International Journal of Computational Intelligence, 3*(2), 104–110.

Laleh, N., & Abdollahi Azgomi, M. (2009). A Taxonomy of Frauds and Fraud Detection Techniques. In S. K. Prasad, S. Routray, R. Khurana & S. Sahni (Eds.), *Information Systems, Technology and Management* (Vol. 31, pp. 256-267): Springer Berlin Heidelberg.

Li, J., Huang, K. Y., Jin, J., & Shi, J. (2008). A survey on statistical methods for health care fraud detection. *Health Care Management Science, 11*(3), 275-287.

Lin, J. H., & Haug, P. J. (2006). *Data preparation framework for preprocessing clinical data in*

data mining. Paper presented at the AMIS Annual Symposium Proceedings.

Major, J. A., & Riedinger, D. R. (2002). EFD: A Hybrid Knowledge/Statistical Based System for the Detection of Fraud. *Journal of Risk and Insurance, 69*(3), 309-324.

Negash, S. (2004). Business intelligence. *The Communications of the Association for Information Systems, 13*(1), 54.

Nigrini, M. J. (1992). *The detection of income tax evasion through an analysis of digital frequencies.* Dissertation, Cincinnati, OH: University of Cincinnati.

Nigrini, M. J., & Mittermaier, L. J. (1997). The use of Benford's Law as an aid in analytical procedures. *Auditing, 16*, 52-67.

NVHHS. (2009). *DME Information Sheet*. (NMO 1115E 11/09). Carson City, NV: Retrieved from http://dhcfp.nv.gov/pdf%20forms/FactSheets/1115E.pdf.

Ortega, P. A., Figueroa, C. J., & Ruz, G. A. (2006). *A medical claim fraud/abuse detection system based on data mining: a case study in Chile.* Paper presented at the International Conference on Data Mining, Las Vegas, NV.

Rosacker, K. M., & Olson, D. L. (2008). Public sector information system critical success factors. *Transforming Government: People, Process and Policy, 2*(1), 60-70.

Ryan, K. (2011). Weighing the Impact of Cuts: Social Security, Medicare and Medicaid. *Public News Service*. Retrieved from News in the Public Interest website: http://www.publicnewsservice.org/index.php?/content/article/21180-1

Sokol, L., Garcia, B., Rodriguez, J., West, M., & Johnson, K. (2001). Using data mining to find fraud in HCFA health care claims. *Topics in health information management, 22*(1), 1-13.

Vann, J. L. (2004). Resistance to Change and the Language of Public Organizations: A Look at "Clashing Grammars" in Large-Scale Information Technology Projects. *Public Organization Review, 4*(1), 47-73. doi: 10.1023/B:PORJ.0000015651.06417.e1

Wang, J., & Yang, J. G. S. (2009). Data Mining Techniques for Auditing Attest Function and Fraud Detection. *Journal of Forensic & Investigative Accounting, 1*(1).

Watrin, C., Struffert, R., & Ullmann, R. (2008). Benford's Law: an instrument for selecting tax audit targets? *Review of Managerial Science, 2*(3), 219-237.

Wegener, D., & Rüping, S. (2010). On Integrating Data Mining into Business Processes. In W. Abramowicz & R. Tolksdorf (Eds.), *Business Information Systems* (Vol. 47, pp. 183-194): Springer Berlin Heidelberg.

Wickizer, T. M. (1995). Controlling Outpatient Medical Equipment Costs Through Utilization Management. *Medical care, 33*(4), 383-391.

Wilkin, C., & Riddett, J. (2009). IT governance challenges in a large not-for-profit healthcare organization: The role of intranets. *Electronic Commerce Research, 9*(4), 351-374. doi: 10.1007/s10660-009-9038-0

Yang, W. S., & Hwang, S. Y. (2006). A process-mining framework for the detection of healthcare fraud and abuse. *Expert Systems with Applications, 31*(1), 56-68.

Editor's Note:

This paper was selected for inclusion in the journal as a CONISAR 2011 Distinguished Paper. The acceptance rate is typically 7% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2011.

Appendix

Table A-1. Summary of Fraud Detection Methods Relevant to Healthcare

Detection Method	Description	Benefits	Drawbacks	Key Findings from Prior Research
Auditing	Medical experts review individual claims one-by-one. Claims are usually selected by a random sample, but could be a targeted sample. Relies on human expertise.	Accuracy, Comprehensive	Costly, time consuming, requires experienced personnel, inefficient	• Found that the best sampling method depends on what is being measured (Buddhakulsomsiri & Parthanadee, 2008)
Statistical: Supervised	Medical and claims experts identify known fraudulent and known honest claims. These claims are modeled to forecast unknown claims. Uses BI data mining tools such as neural networks and fuzzy logic	Proven technology in business fraud. Quickly pinpoints suspicious providers. Widely used.	Cannot detect new types of fraud. May identify legitimate claims as fraudulent. Requires expertise prior to detection of fraud. Requires knowledge of complex BI tools.	• Created sub-models with feedback connections (Ortega, et al., 2006) • Determined that two categories were more productive than four. (He, et al., 1997)
Statistical: Unsupervised	Statistical algorithms are used to identify outliers based on pre-defined categories. Filters out anomalous behavior from peer groups. Anomalous data is examined by claims experts to detect. Uses BI statistical tools such as standard T-tests, correlation, clustering, and regression.	Quickly pinpoints suspicious providers. Can detect new types of fraud.	May identify legitimate claims as fraudulent. Requires examination of claims after statistical evaluation. Requires knowledge of statistical methods.	• Identified key categories for health care fraud (Major & Riedinger, 2002) • Recommended use of clustering in data mining (Bolton & Hand, 2002) • Found Benford's Law applicable for fraud detection (Nigrini, 1992); (Nigrini & Mittermaier, 1997)

Table A-2. Top Counts of Flagged Suppliers

	Feature Variable													
Supplier ID	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**Number of times flagged**	**Total Net Pay Amt**
179	1	1	0	1	0	0	1	1	0	1	1	1	8	$146,160
100	1	1	0	1	0	0	1	1	0	1	1	0	7	$215,586
307	1	1	0	1	0	0	1	1	0	1	1	0	7	$33,930
303	0	1	0	1	0	1	0	1	0	1	1	0	6	$783
88	1	0	0	1	0	0	0	1	0	1	1	0	5	$44,053.5
192	1	0	0	0	0	0	1	1	0	1	1	0	5	$8,587