INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Improving Web-Catalog Design for Easy Product Search

I. Robert Chiang, Manuel A. Nunez,

To cite this article:
I. Robert Chiang, Manuel A. Nunez, (2007) Improving Web-Catalog Design for Easy Product Search. INFORMS Journal on Computing 19(4):510-519. https://doi.org/10.1287/ijoc.1060.0184

INFORMS Journal on Computing
Vol. 19, No. 4, Fall 2007, pp. 510–519
ISSN 1091-9856 | EISSN 1526-5528 | 07 | 1904 | 0510

DOI 10.1287/ijoc.1060.0184

Improving Web-Catalog Design for Easy Product Search

I. Robert Chiang
Accenture, Hartford, Connecticut 06103, robert.chiang@accenture.com

Manuel A. Nunez
School of Business, University of Connecticut, Storrs, Connecticut 06269, manuel.nunez@business.uconn.edu

Building intuitive web sites is important for online businesses as positive experiences toward the virtual storefront could translate to customer goodwill and repeat visits. Streamlining web-site navigation is further motivated by the availability of comprehensive site-visit traces such as shopping carts and click-stream logs. We incorporate such sales and browsing patterns for the autonomous design of web catalogs by reducing the expected click count it takes to find related items. We first model catalog design as a task of placing items onto catalog pages, leading to a quadratic assignment formulation. For a more extensive redesign, such as when the number and location of links on each page are also decision variables, we propose a two-stage optimization procedure to ensure that click-count reduction is not achieved at the cost of excessive page cluttering. Our analysis reveals that an optimized catalog is robust against shifts in browsing behavior and that simplistic interfaces are preferred for users at either end of the experience spectrum. Using genetic algorithms, we find catalog designs that significantly outperform those found with a traditional greedy heuristic.

Key words: catalog design; network design; quadratic assignment problem; genetic algorithms; electronic commerce

History: Accepted by Prakash Mirchandani, Area Editor for Telecommunications and Electronic Commerce; received July 2004; revised June 2005, March 2006; accepted March 2006. Published online in *Articles in Advance* July 25, 2007.

1. Introduction

Following the "irrational exuberance" fueled by speculation toward the commercialized Internet, many of the dubbed "dot goners" have turned out to be thriving and profitable enterprises (Economist 2002, Mullaney 2004). Many key indicators, including the volume of online B2B/B2C transactions (Hoffman et al. 2002) and the e-business' contribution to productivity (Mullaney et al. 2003), have either met or eclipsed bold pre-bubble predictions. With over ten percent of the worldwide population having gained Internet access (Nua Internet Surveys 2003), e-commerce will play an ever-increasing economic role.

Many online retailers have experienced poor order fulfillments (Rosen 2001), overwhelmed or crashed servers, and sloppy control of sensitive data. Innovations such as supply-chain alliances (Bhargava et al. 2006), server optimization (Tan et al. 2005), and data-security enhancement (Muralidhar et al. 2001). While back-end blunders have dwindled, poor web design remains a source of customer aggravation, resulting in missed business opportunities (Tedeschi 2003) and low browsing-to-sales conversions (Hurst 1999), as customers were not aware of or had difficulty locating desired products or services.

A virtual storefront should be engaging and intuitive. Web-site design thus involves psychological considerations such as content richness and aesthetics (Aladwani and Palvia 2002, Hong et al. 2004). A design consideration is effective placement of hyperlinks, an issue especially critical for web catalogs (Mathwick et al. 2001). The catalog should enable customers to reach pages of interest quickly and provide smooth transition toward pages deemed "related."

Catalog navigation has typically been category/subcategory hyperlinks on the side bar. For example, one would click a sequence like "Electronics > Camera & Photo > Digital Camera > 5 Megapixel or higher." Alternatively, one can submit "6MP digital camera optical zoom" to the search engine for a list of recommendations. Relying on a category hierarchy or a search engine is "passive" navigation, querying the database with intrinsic attributes.

A shortcoming of passive hyperlinks is that to view another item several "hops" may have to be traversed. To ensure good coverage, a search engine may lead to many recommendations, which discourages further exploration (Tan et al. 2004). A search engine also interleaves two input devices, which could disrupt the "flow" of catalog browsing (Schaffer and Sorflaten 2004).

To make navigation more intuitive, web sites have adopted "Related Items" or "You May Also Like" sections in their catalog for a quick click-through. Many products can be considered in such a section, from substitutable to complementary. The issue is to recognize and reflect how items are correlated in the catalog. While item placement could be manual, relying on human expertise, going beyond a boutique would require a more systematic approach.

We incorporate consumer browsing and purchasing behaviors for automation of web-catalog design. Mining financial and operational data has improved business decisions, such as more precise expansion planning (Schlosser 2003) and better logistics efficiency (Brown 2002). Browsing behavior has also been used for advertisement design (Karuga et al. 2001) and customer relationship management (Padmanabhan and Tuzhilin 2003). The navigation-design issues studied here allow for cross-product recommendation in collaborative filtering (Anderson et al. 2003, Sarwar et al. 2001) and ensures certain topological constraints.

We develop a catalog-recommendation system. While the top menu or side bar can search for specific products, we concentrate on the "Related Items" links to facilitate browsing and impulse buying. We model the catalog as a network of nodes (catalog pages) and edges (hyperlinks). Under consideration are three key factors that affect catalog navigation: item correlations, catalog topology, and user experience. How items correlate with can be observed since web servers capture site-visit and transaction histories. Naturally, items that are often jointly purchased or evaluated should be placed "close" to each other.

Catalog topology describes where and how densely catalog pages are connected by the hyperlinks. To see how topology affects navigation, a fully connected catalog renders one-click access to all other pages, but severely clutters, defeating the usefulness of "quick" links. If an item relates little with the rest, few links should stem from its page; if an item is often viewed or purchased with many others, more links may be preferred even if doing so increases local page cluttering.

User experience reveals the effect of limited (page and link) visibility of browsing. A first-time visitor may find it more difficult to locate items than returning customers, thus incurring more intricate page traversals. Traditional design methods do not account for user experience. By not offering a customized catalog, the web site could miss opportunities.

We suggest how an online storefront can be adapted for different customer segments. By analyzing the traverse log and topology of a catalog, we propose a procedure to extract the browsing behavior of customers and identify the best catalog for customers at each experience level. Two approaches accommodate the difference in user experience: a generic approach in which one catalog is designed to match the average browsing behavior of all users, and a customized approach in which several catalogs are generated and dynamically assigned according to the experience level. The tradeoff between the two approaches depends on whether the improved ease of navigation justifies the cost of generating/managing multiple catalogs. The same procedure can be applied toward design of customized catalogs based on other segmenting attributes.

We also propose two optimization models for catalog navigation depending on whether placement of hyperlinks is to be determined. The first model applies when the topology is given, such as when a web site uses a fixed number of hyperlinks on each page. If so, we find the best assignment of items onto catalog pages, using a normalized quadratic assignment problem. The second optimization model takes place when the catalog topology is also subject to change. Since the placement of hyperlinks needs to be determined before the pair-wise distances between pages can be known, a two-stage approximation is proposed: a genetic-algorithm (GA) heuristic to search for an optimal catalog topology, and a second GA is used by the first to determine near-optimal item assignment. Numerical results suggest that our procedures generate near-optimal catalog designs and consistently outperform traditional greedy, polynomial-time assignment heuristics by a significant margin. In some cases, the catalog, once optimized, is quite robust against shifts in average user browsing behavior, so few customized catalogs would serve a wide range of users. A more sparse topology is favored with novice users. Our approach leads to an autonomous and efficient design process that could help generate customized or personalized catalogs.

Finally we suggest how to incorporate sequences of more than two items for product recommendation. By considering longer chains of products already viewed in the current browsing session, it is possible to capture more complex shopping behavior than a system based on using the latest item alone. The chain-based formulation leads to a general polynomial assignment/design problem and, in some cases, the objective function can be reduced to a simpler quadratic form.

Considering both the catalog topology as well as shopping frequencies facilitates better capturing of higher-order correlations than does a more traditional myopic approach. Figure 1 shows two different layouts for a catalog with four pages (1–4) and four items (A–D). The pairwise purchasing frequency matrix of the items is in Table 1. Entry 15% in the matrix was computed by determining the total number of pairs

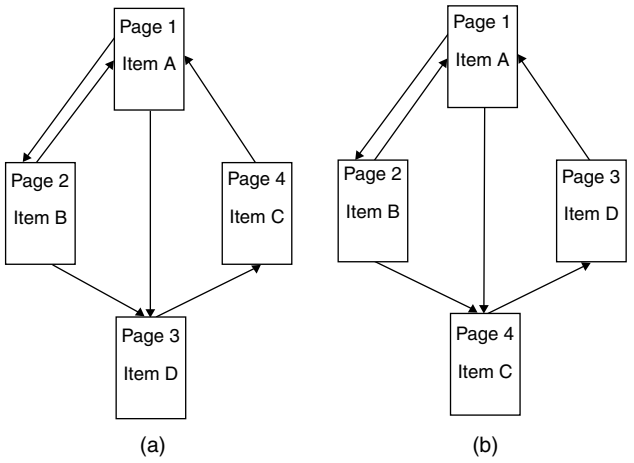

Figure 1 Comparing Designs

of items consecutively purchased or viewed in order from historical data, and then taking the percentage that items D and C (in that order) appear among all the pairs of items. All other entries were similarly computed.

Using a greedy approach based on the highest frequency and starting by assigning item A to page 1, we would link items B and C to item A, obtaining the layout in Figure 1(a). From the average click-count formula described later, we obtain an average count of 1.68 clicks for the catalog in Figure 1(a). On the other hand, the average count for the optimized layout in Figure 1(b) is 1.42 clicks. Given that the improvement is 18% in such a small scale, it is easy to appreciate the potential gains for larger instances.

Section 2 defines the pair-wise distance between pages and formulates an assignment optimization model. The solution procedure and numerical results are then presented. The design-optimization problem that aims to improve the topology is in Section 3, followed by discussion of a solution procedure and numerical results.

In the Online Supplement to this paper on the journal's website, Section 1 discusses the tradeoff between generic and customized network designs. Section 2 of the Supplement discusses the path-based recommendation model and the corresponding polynomial assignment problem. The last two sections of the Supplement describe the greedy heuristics for both the assignment and design-optimization problems.

Table 1 Shopping Frequencies

Item	A	B	C	D
A	0	10%	10%	7%
B	10%	0	10%	7%
C	12%	12%	0	1%
D	3%	3%	15%	0

2. The Assignment Model

We consider a situation in which some "design cues" are copied from the existing catalog. To be visually consistent, a designer may not alter the number of thumbnails in the "Related Items" section when the catalog is undergoing changes. In such a case, we show that navigation design can be modeled as a quadratic assignment problem.

2.1. Distance Between Pages

From logs of past visits, it is possible to recognize "clustered" items that are frequently viewed or purchased within a session of user visits and should in close proximity. It is thus necessary to measure the distance between pages prior to item placement. We represent the topology as a directed network $G = (N, E)$ with $|N| = n$ nodes and $|E| = m$ arcs (hyperlinks). Figure 2 shows a simple topology with $n = 7$ pages and $m = 12$ links.

Let D_{ij} equal distance between pages i and j. Distance can be measured in many ways and it depends on the experience of the user in using the catalog. An experienced user who knows very well the network topology would tend to use the shortest path (fewest clicks) between pages. Hence, we use a matrix of shortest distances between pages as a proxy for the behavior of an experienced user. The shortest distances can be obtained by performing a breadth-first search. For a catalog with n nodes, the algorithm is $O(n^2)$ (Ahuja et al. 1993, Cormen et al. 2000). Let the matrix D^E represent the minimal click counts between pages. In the catalog of Figure 2,

$$D^E = \begin{bmatrix} 0 & 1 & 1 & 2 & 2 & 2 & 2 \\ 1 & 0 & 2 & 1 & 1 & 3 & 3 \\ 1 & 2 & 0 & 3 & 3 & 1 & 1 \\ 2 & 1 & 3 & 0 & 2 & 4 & 4 \\ 2 & 1 & 3 & 2 & 0 & 4 & 4 \\ 2 & 3 & 1 & 4 & 4 & 0 & 2 \\ 2 & 3 & 1 & 4 & 4 & 2 & 0 \end{bmatrix}. \quad (1)$$

(1) shows the minimum number of pages traversed from source to destination. For instance, the distance between page 1 and page 7 is 2 (upper right corner). We assume zero distance between a page and itself.

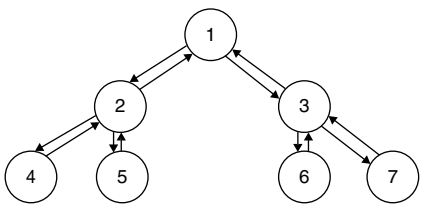

Figure 2 Binary Tree Network

Since a web interface permits only limited visibility, page traversing would likely incur higher click counts than in D^E. In particular, an inexperienced user would tend to browse the site randomly. Since there is no reason to choose one page over another, a novice might traverse the catalog via a uniform Markovian random walk. The resulting distance matrix D^N for the average novice user is thus the first passage time from a source page to the destination. To compute D^N, let P be the matrix of one-step transition probabilities for the corresponding Markov chain. If node k in the network has d ($d > 0$) outgoing hyperlinks, we have

$$P_{kl} := \begin{cases} \dfrac{1}{d} & \text{if } (k,l) \in E, \\ 0 & \text{if } (k,l) \notin E, \end{cases}$$

for all $k \in N$. For Figure 2,

$$P = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 1/3 & 1/3 & 0 & 0 \\ 1/3 & 0 & 0 & 0 & 0 & 1/3 & 1/3 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

To compute D^N we can solve n linear systems in $O(n^3)$ time (Golub and Van Loan 1997). Because of the definition of P, the Markov chain is irreducible. Let P^{l-} be a matrix identical to P except that zeros replace the l-th row, and let e^{l-} be the all-ones vector except that a zero replaces the l-th entry. If D_l^N is the l-th column of the distance matrix D^N, D_l^N can be computed by solving $D_l^N = e^{l-} + P^{l-}D_l^N$ for all l. For Figure 2,

$$D^N = \begin{bmatrix} 0 & 7 & 7 & 14 & 14 & 14 & 14 \\ 5 & 0 & 12 & 9 & 9 & 19 & 19 \\ 5 & 12 & 0 & 19 & 19 & 9 & 9 \\ 6 & 1 & 13 & 0 & 10 & 20 & 20 \\ 6 & 1 & 13 & 10 & 0 & 20 & 20 \\ 6 & 13 & 1 & 20 & 20 & 0 & 10 \\ 6 & 13 & 1 & 20 & 20 & 10 & 0 \end{bmatrix}.$$

Random-walk behavior occurs in other search or browsing patterns. In peer-to-peer networks (Adamic et al. 2001) the name of a target file may be known, but the node holding the file is not known prior to a real-time search. A random-walk search is efficient in such cases. See also Deo and Gupta (2001, 2003), and Lv et al. (2002).

To approximate the actual click count of a typical user, we use the convex combination of the two distance matrices D^E and D^N:

$$D(\theta) := \theta D^E + (1-\theta)D^N, \quad 0 \le \theta \le 1. \tag{2}$$

Choice of θ (the *experience index*) would depend on a user's experience and the intended use of the catalog. A B2B site getting routine purchases could expect a high θ, but a catalog should be optimized for low θ if designed for a general store encouraging visitors to explore different "aisles." Section 1 in the Supplement provides further detail on choosing θ.

We assume without loss of generality that the network is connected, i.e., there is always a directed path connecting two pages. Otherwise, if not every item is reachable by all others through the recommendation system ("related items" or "you may also like" links), it is always possible to go back to the root node (home page) by using the "back" button, the "home page," a main menu, etc. Hence, distances between pages are always finite, perhaps by adding an artificial dummy node representing a step "out" of the recommendation network and penalizing such a step with a large click count. It is not difficult to adapt the computation of the distance matrices D^E and D^N to account for the click penalty of traveling through the artificial node. Alternatively, the designer could cluster items based on frequencies, using statistical or machine-learning techniques, and then solve separate design problems assuming a connected network for each cluster.

2.2. Assignment Problem

Item assignments should be influenced by how they relate to each other. To capture such a correlation, we use an $n \times n$ frequency matrix F to show past shopping patterns and browsing behaviors. An entry F_{ij} (≥ 0 for all i, j) represents the proportion of occurrences in which the pair of items (i, j) appeared in a collection of shopping carts or was evaluated in a collection of comparison-shopping sessions. We have

$$\sum_{i,j} F_{ij} = 1. \tag{3}$$

To record the page to which each item is assigned, we use an $n \times n$ assignment matrix X, where X_{ik} is 1 if item i is assigned to page k, 0 otherwise. Given a distance matrix D and a frequency matrix F, we model the process of improving catalog navigation as finding the optimal assignment matrix X^* that solves:

$$\begin{aligned} \text{(AP)} \quad \min \quad & z(X) = \operatorname{trace}(FXDX^T), \\ \text{s.t.} \quad & Xe = e, \\ & X^T e = e, \\ & X_{il} \in \{0,1\}, \quad \forall i, l; \end{aligned}$$

where e denotes the n-dimensional all-ones vector. How items are assigned to pages (i.e., the assignment matrix X) influences the objective function. In (AP), $z(X)$ is the average number of clicks it takes to find a pair of items of interest for a given X. The constraints ensure that one and only one item is assigned to each page.

To study the complexity of (AP) notice that, except for condition (3), it is quadratic assignment problem (QAP) in Koopmans and Beckmann (1957) form, so any instance of QAP with arbitrary (rational) cost matrices F and D can be polynomially transformed to an instance of (AP) by normalizing the cost matrix F. Since the QAP is *NP*-hard (Garey and Johnson 1979), (AP) is too, so we use a GA-based heuristic to find approximate solutions. Section 2.4 shows computational results.

2.3. GA Approach for the Assignment Problem

Care must be taken when using GA in constrained optimization, as it could be costly to repair infeasible solutions after applying standard GA operations such as crossover and mutation. We use a chromosome representation consisting of permutation vectors where every feasible solution to (AP) is represented as an n-dimensional vector v whose v_i is the page to which item i is assigned. In Figure 2, a possible chromosome is $v = (2, 5, 1, 7, 6, 3, 4)$. In this example $v_3 = 1$ means that item 3 is assigned to page 1. The total number of possible chromosomes/feasible solutions is $n!$, so the search space grows quickly with n.

There are several standard permutation-preserving crossover operations available (e.g., IDX and LOX in Chambers 1995). We use a variant called *median crossover* (MDX) (Drezner 2003, Drezner and Salhi 2002) for its ability to combine crossover with local search for permutation chromosomes. In MDX, the network is divided into two cohesive parts according to the median distance of the network nodes with respect to a fixed pivot node. Then, offspring are generated by taking the segment under the median from one parent and the segment over the median from the other parent. In Figure 3 there are five node-pages and the pivot is node 1. Using shortest-path distances, the numbers next to each node represent the distance with respect to the pivot. The median distance is 2 and the nodes with distance below or equal to the median are 1, 2, and 4. Given two parent chromosomes $v = (3, 4, 1, 5, 2)$ and $\hat{v} = (5, 1, 4, 2, 3)$, we generate one descendant by taking the assignments from v corresponding to nodes below the median to obtain $(x, 4, 1, x, 2)$ and the we fill the remaining empty spots with assignments from $\hat{v}$ to yield $(5, 4, 1, 3, 2)$. Similarly, from $\hat{v}$ we obtain $(x, 1, 4, 2, x)$ and then, using v, we yield a second descendant $(3, 1, 4, 2, 5)$. Since a crossover depends on which node is chosen to be pivot, a local search is performed in which only the two best offspring from all the possible pivot crossovers are selected. Following Drezner (2003), we also implemented a post-merging procedure (PMP), based on a simple Tabu search, which takes one of the descendants from the crossover and checks the change in objective value for all pairwise exchanges of pages in the chromosome. If an improving exchange is found, then the best exchange is executed and the process is repeated.

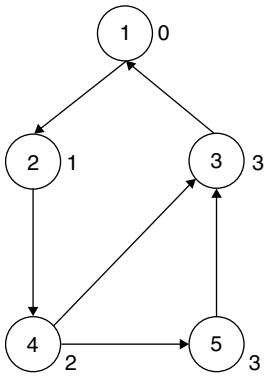

Figure 3 An Example of MDX

As for the mutation operator, and to keep mutated chromosomes feasible, we use random swapping of the assignment of two items in a given chromosome vector or the PMP described above. Drezner (2003) showed that a GA based on the MDX crossover and PMP not only produces near-optimal solutions, but is also fast, and that this GA performs well compared with other standard QAP algorithms. Finally, we use elitism (Michalewicz 1994) to retain best-candidate solutions across generations.

To compare the performance of our GA we use a greedy heuristic to find approximate solutions to (AP), based on traditional recommendation methods in which, once an item has been assigned to a page, other items that have a high probability of being purchased together with the item are directly linked to that page. Details are in the Supplement.

2.4. Numerical Results

We start with the task of re-designing a small catalog. A group of 12 items was extracted from the web site of a major home-improvement center. Figure 4 illustrates the original catalog topology. To populate the frequency matrix F we first form clusters among items and then assign high frequency values for items that are in the same cluster. We also assign low frequency values for items across clusters. Table 2 describes the items (as numbered in Figure 4) as well as the clusters. The clusters are formed according to intrinsic product characteristics: Cluster A is light bulbs, Cluster B is household tools, Cluster C is electrical wiring

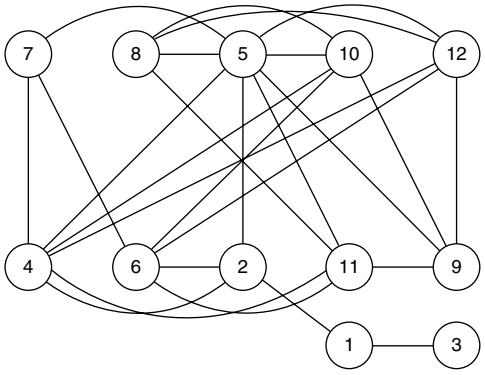

Figure 4 **A Catalog Topology for Electrical and Lighting Tools and Supplies**

supplies, and Cluster D is electrical wiring tools. For example, between a fish tape and a fish-tape leader we set a weight of 10 because they are in the same cluster, whereas between an indoor timer and an electrical tape we set a weight of 1 because they are in different clusters. We obtain F satisfying (3) by dividing each weight by the sum of the weights.

Table 3 compares the objective values obtained from different methods. The objective-value column contains solutions obtained by complete enumeration. The expected click counts range from 2 to more than 8. In each case, a local-search GA was able to identify the optimal item assignment by evaluating only 2,000 (a population size of 20 times 100 generations) among the $12! \approx 4.7 \times 10^8$ possible assignments. The GA outperforms the greedy heuristic by at least 37 percent. Using a 3.2 GHz Pentium 4 processor and 1 GB RAM, both GA and the greedy heuristic take only several seconds, whereas the complete enumeration takes more than four hours.

To ensure that the solution time and quality for the GA scales well, we randomly generated a 120-node test problem with 10 categories and 12 products in each category; we randomly assigned weights of either 5 or 10 to pairs of elements within the same category. We assigned 1 to pairs of elements across categories. Then, we normalized the weights by dividing each weight by the sum of weights to obtain the frequency matrix. Finally, we assumed a three-link-per-page topology. We increased the GA population size to 100.

Table 2 **Items Description**

Item	Description	Cluster	Item	Description	Cluster
1	Floodlight bulbs	A	7	Tape rolls	C
2	Bulb changer kit	B	8	Vinyl tape	C
3	Indoor timer	B	9	All purpose tool	D
4	Electrical tape	C	10	Fish tape	D
5	Wire connector kit	C	11	Fish tape leader	D
6	Cable tie kit	C	12	Scissors with stripping	D

Table 3 **Click-Count Performance**

Method	θ	Objective value (Z)	Relative error (%)
Enumeration	0.5	8.324	N.A.
	0.9	2.775	N.A.
	0.95	2.082	N.A.
GA	0.5	8.324	0.00
	0.9	2.775	0.00
	0.95	2.082	0.00
Greedy heuristic	0.5	12.233	46.97
	0.9	3.886	40.02
	0.95	2.843	36.55

Table 4 gives the results. Complete enumeration is not possible within a reasonable time frame. GA on average generates an objective value that is 10.4% lower than the greedy solution if the number of generations is set at 1,000. If the number of generations reaches 5,000, the GA solution on average is 24.4% lower than the greedy heuristic, closely matching Table 3 for the 12-node case. However, the execution time for the GA increases with the number of generations. While the GA time is comparable to the greedy heuristic, at several minutes when the number of generations is 1,000, increasing the number of generations lengthens the execution time proportionally.

Finally, we study the convergence of the GA to optimality. We generated a sample of 20 problems, each consisting of 60 nodes/pages. Each problem was generated to create a special directed ring topology that includes all arcs $(1, 2)$, $(2, 3)$, $\ldots$, $(59, 60)$; arc $(60, 1)$; and arcs of the form $(1, 30)$, $(2, 31)$, $\ldots$, $(31, 60)$. Frequency matrices were randomly generated to ensure that one of the item cyclic permutations $(1, 2, \ldots, 60)$, $(2, 3, \ldots, 60, 1)$, $\ldots$, $(60, 1, 2, \ldots, 59)$ is the optimal solution. Hence, it is only necessary to evaluate 60 solutions to determine the optimal solution. We ran our GA and the greedy heuristic on the sample problems for $\theta = 0.9$. Each GA run was executed a maximum of 500 iterations with a population size of 100. Figure 5 compares the average objective value of the best solution per iteration as well as the average heuristic objective value and the average optimal value.

Figure 6 shows the percentage of sample problems for which the GA best solution per iteration is within 1% of the optimal solution. By iteration 500 the GA is within 1% of the optimal solution for 18 out of 20

Table 4 **Click Count Improvement over Greedy Heuristic**

θ	GA (1,000 iterations) (%)	GA (5,000 iterations) (%)
0.5	9.18	21.45
0.9	9.35	29.06
0.95	12.61	22.57

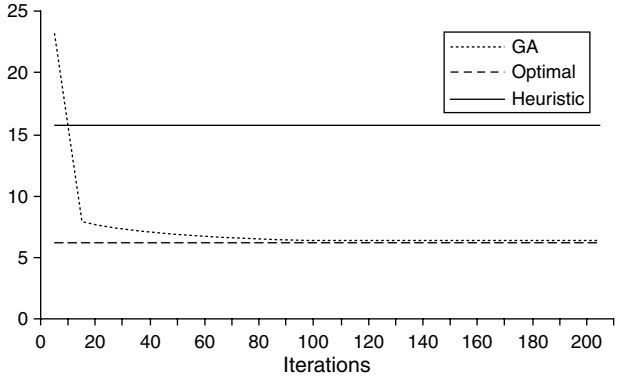

Figure 5 Convergence of the GA

of the sample problems. For the remaining two problems, the GA is within 5% of the optimal solution by iteration 500.

3. The Design Model

Although the assignment model discussed in the previous sections significantly improved catalog navigation, it operates under a restricted catalog topology. For some web sites, the number of hyperlinks is not fixed but depends on which item is assigned to the catalog page. When catalog topology is also a decision variable, the main concern is to ensure that page links would provide sufficient connectivity without incurring excessive cluttering.

3.1. Model Description

To model how catalog pages are connected we use a node-node adjacency matrix Y to represent the placement of hyperlinks. We set Y_{kl} to 1 if page k and page l are directly connected by a network edge, and 0 otherwise. We assume that Y is chosen such that the resulting network is connected. For a given Y, we define the weighted distance matrix similarly to (2) as

$$D(Y) := \theta D^E(Y) + (1-\theta)D^N(Y), \quad 0 \le \theta \le 1,$$

where $D^E(Y)$ and $D^N(Y)$ are the experienced and novice distance matrices induced by the topology Y, respectively.

In addition to placement, the number of links affects navigation. While too few links may make parts of the catalog difficult to reach, too many links creates page cluttering and makes the catalog confusing to traverse. We denote by $C(Y)$ the cluttering cost. For simplicity, we assume that $C(Y)$ is proportional to the number of links in the network, or $C(Y) = c\sum_{k=1}^{n}\sum_{l=1}^{n} Y_{kl}$, where c is a positive constant. The form of $C(Y)$ is consistent with literature on cognitive design of user interfaces, which suggests a linear (or logarithmic) relationship between evaluation time and the number of choices on the page (e.g., Norman 1991).

The design optimization model is

$$\begin{aligned}
\text{(DP)} \quad \min \ & z(X, Y) = \text{trace}(FXD(Y)X^T) + C(Y), \\
\text{s.t.} \ & Xe = e, && (4)\\
& X^T e = e, && (5)\\
& \mathcal{N} V^k = b^k, \quad \forall k, && (6)\\
& \sum_{k=1}^{n} V^k \le n(n-1)Y, && (7)\\
& V^k \ge 0, \quad X_{il}, Y_{kl} \in \{0,1\}, \quad \forall i, k, l.
\end{aligned}$$

In (DP), (4) and (5) are the assignment constraints as in (AP), and (6) and (7) ensure connectivity as follows. Assume there are $n-1$ units of a commodity to be shipped from a node, and there are demands of 1 at all other nodes. If the commodity originating from node k is sent via the edge between page i and page j, then V_{ij}^k is the flow through edge (i, j). (6) gives the flow-balance equations for each commodity, where $\mathcal{N}$ is the node-arc incidence matrix, and b^k is the demand-supply vector at node k. Because the flow through each arc is at most $n-1$ units for each commodity, $n(n-1)$ is an upper bound for the total flow of commodities through edge (i, j). Hence, (7)

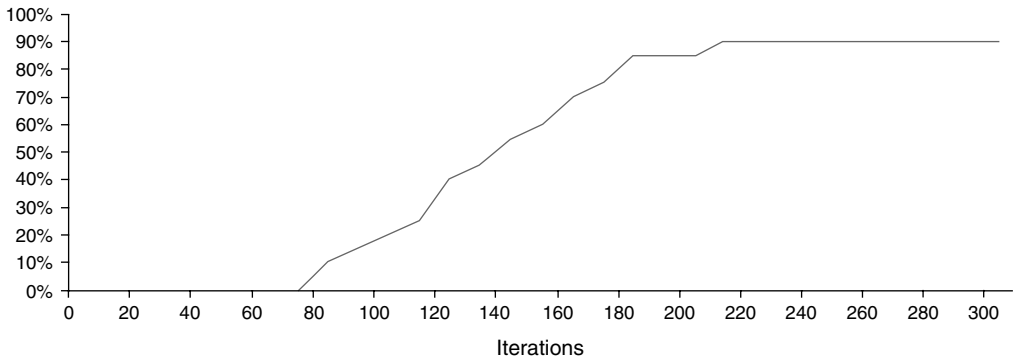

Figure 6 Percentage of Sample Problems for Which the GA Converged

implies that $Y_{ij}=1$ if arc (i,j) is used to ship at least one unit of a commodity, and that there is no flow through arc (i,j) if $Y_{ij}=0$.

Because solving (DP) requires identifying the best item assignment for each topology being evaluated, (AP) is a subproblem of (DP), so (DP) is *NP*-hard. We propose a solution procedure that involves two GAs in Section 3.2.

The choice of c in the cluttering cost $C(Y)$ could be done parametrically, i.e., different optimal (or near-optimal) designs will be found for corresponding different values of c. Or, $C(Y)$ could be removed from the objective function and a new constraint of the form $\sum_{k=1}^{n}\sum_{l=1}^{n} Y_{kl} \le L$ could be added to find an optimal design with no more than L total links; or even further, add a set of constraints of the form $\sum_{l=1}^{n} Y_{kl} \le L_k$ to restrict the number of links on page k to L_k. It is not difficult to modify the GA proposed in Section 3.2 to adjust for these variants.

3.2. GA

To find approximate solutions to (DP) we use a GA to search in the space of connected network topologies. The chromosome representation of each feasible solution in the search space consists of a binary matrix Y representing the node-node adjacency matrix of a connected network. Similar to the assignment model, we use a tweaked median crossover MDX that has proven to be very good for network design problems (Drezner and Salhi 2002). We also use elitism across generations and a connectivity-repair procedure for the mutation operator.

Since the constraints in (DP) are separable, we can re-write it as

$$\min\left\{C(Y)+F(Y):\ \mathcal{N}V^k=b^k,\forall k,\ \sum_{k=1}^{n}V^k\le n(n-1)Y,\ V^k\ge 0,\ Y_{kl}\in\{0,1\},\forall k,l\right\},$$

where

$$F(Y) := \min\{\text{trace}(FXD(Y)X^T):\ Xe=e,\ X^Te=e,\ X_{il}\in\{0,1\},\ \forall i,l\}.$$

Hence, the fitness/evaluation of a given chromosome Y consists of the cluttering cost $C(Y)$ plus the click-count cost $F(Y)$ of the optimal assignment in the corresponding network. Because computing $F(Y)$ entails solving (AP), we use a second GA (from Section 2.3) to find an approximate solution to the optimal assignment. Therefore, we apply a "master-subproblem" procedure in which a master GA for the design problem uses (AP) GA as a subroutine to estimate the fitness of each chromosome in a given population.

Table 5 GA Improvement in the Objective Value

	Cluttering cost $c=1$			Cluttering cost $c=10$		
θ	Initial pop.	Final pop.	% Diff. (%)	Initial pop.	Final pop.	% Diff. (%)
0.95	30.847	13.842	55.13	252.048	112.918	55.20
0.9	27.464	15.055	45.18	272.486	114.100	58.13
0.5	35.615	21.892	38.53	237.316	122.384	48.43

3.3. Numerical Results

Only the numerical results for the 12-node design problem are reported. The population size and the number of generations are set at 20 and 100, respectively, for both the (DP) GA and the (AP) GA. To be consistent, we populate the frequency matrix F as described in Section 2.4.

Table 5 shows a significant difference in objective values between the initial and final GA populations. Navigation could deteriorate severely by a poor choice of topology. Not surprisingly, the GA final population objective value increases as the experience index θ decreases or as the cluttering-cost coefficient c increases.

Figure 7 shows the best designs at two different levels of cluttering cost. When the penalty for page cluttering is less severe, having more hyperlinks is preferred, leading to the 15-link topology of Figure 7(a). Due to the nature of the frequency matrix F, items are also perfectly clustered (i.e., items inside each cluster are directly connected). On the other hand, when c becomes larger, Figure 7(b) shows that the topology is reduced to a tree structure with a 27% reduction in the number of hyperlinks. Also, items are no longer perfectly clustered with large c, potentially due to the highly constrained nature of a tree topology. Unlike solving (AP) for a 12-node problem that takes only a few seconds, it takes on average 30 minutes to solve the corresponding (DP).

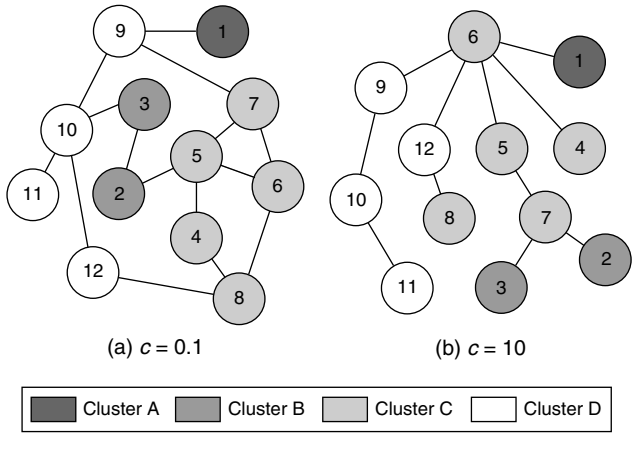

Figure 7 The Impact of Page Cluttering on Catalog Topology

Table 6 Comparing a Greedy Heuristic to the GA Approach

θ	Average improvement (%)
0.5	87.82
0.9	49.98
0.95	61.85

For (AP) we used a traditional greedy heuristic to compare to the performance of our GA approach, which can be easily extended to (DP). The idea is to produce a network topology in which the number of links per page is constrained. For example, the designer allows only, say, two links per page. The heuristic would assign an item to a free page, and then explore the item's row in the frequency matrix to link the item's page to as many other items as possible according to the frequency order (greedy strategy). This is repeated until all items are assigned to pages (details are in the Supplement). The resulting directed network might not be connected, so either connectivity should be repaired by adding a minimum number of links, or the graph should be treated as in Section 2.

To compare the greedy heuristic to our GA approach, we created a sample of 20 problem instances, each consisting of 30 nodes/pages. The frequency matrix for each problem was randomly generated by drawing numbers with equal likelihood from either a uniform distribution on $[30, 70]$ or on $[80, 120]$, and then normalizing the frequency matrix to satisfy (3). We first ran the heuristic over the 20 problems for three different values of θ to obtain an initial network design and corresponding item assignment to pages for each of the 20 sample problems. Next, we ran the GA for (AP) using the same frequency matrix and network design found by the heuristic. Table 6 shows the average improvement over the greedy heuristic obtained using our GA, at least 50%. Obviously, by running the GA for (DP) the gap between the GA recommended design and the greedy heuristic design will grow.

Table 7 shows the complexity (number of links) of the topology designs (for a fixed clustering cost of 1) as θ varies. For extreme experience factors (close to 0 or to 1) the number of links is reduced. Intuitively, for inexperienced users the greater the number of links the higher the probability of "getting lost" in the random walk, thus yielding higher expected click counts; for experienced users, having more links adds to the page clutter yet helps little to reduce the click count, thus the more simplistic interface.

Table 7 Topology Complexity as θ Varies

θ	Number of links
0.05	11
0.1	14
0.5	13
0.9	11
0.95	11

4. Conclusions

We have proposed two optimization models to improve web catalog navigation. When the topology is given, we show that browsing can be improved by solving a quadratic assignment problem. If the topology is also subject to change, a two-stage optimization procedure is needed to identify improved page connectivity and item assignment. A recently proposed GA that combines standard GA with local search is the optimization heuristic. By comparing to the optimal solution of small problems, the GA obtained high-quality solutions efficiently. When the problem size increases, the GA produced designs significantly better than those produced by a polynomial-time greedy heuristic.

While our models exploit mainly the pair-wise correlation among items, they still are effective in capturing higher-order correlations. For example, if items A, B, and C, are popular and highly correlated, the frequency matrix will show high values between pairs (A, B), (A, C), (B, C), etc., which likely will result in all three items' being clustered together in the optimal design. We also formulated a model to incorporate longer sequences of items (Section 2 in the Supplement). Even though this problem is more complex, under certain circumstances it reduces to a standard quadratic assignment problem.

Although items in the same cluster might be highly correlated, the optimal topology is not necessarily a fully-connected network within the cluster because of cluttering costs. For example, in the 12-node topology in Figure 7(a) there is a star topology within the cluster for items 9, 10, 11, and 12, with item 10 as the center of the star.

We have assumed that the frequency matrix is obtained from actual purchasing or browsing data but one could rely on human expertise to populate the matrix, e.g. when launching a new e-commerce site. Similarly, during the period of product promotion and bundling the designer could use a "supplementary" frequency matrix to overlay the frequency matrix resulting from actual visits, thus boosting correlations among targeted items.

We focused on improving catalog navigation by optimizing hyperlink placement. A fruitful extension might consider a combined design of hyperlinks and the category hierarchy (i.e., the side bar). The goal is to separate items into categories or hierarchy layers, and then use our methods to find the optimal design

for each category or layer. Work is underway on this direction.

Acknowledgments

The authors thank Editor-in-Chief David Kelton, Area Editor Prakash Mirchandani, the associate editor, and three reviewers for their constructive and insightful comments. The authors are also especially grateful to the associate editor for the suggestion concerning the chain-based formulation for catalog navigation design.

References

Adamic, L. A., R. M. Lukose, A. R. Puniyani, B. A. Huberman. 2001. Search in power-law networks. *Physical Rev. E* **64** 046135.

Ahuja, R. K., T. L. Magnanti, J. B. Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Englewood Cliffs, NJ.

Aladwani, A. M., P. C. Palvia. 2002. Developing and validating an instrument for measuring user-perceived web quality. *Inform. Management* **39** 467–476.

Anderson, M., M. Ball, H. Boley, S. Greene, N. Howse, D. Lemire, S. McGrath. 2003. RACOFI: Rule-applying collaborative filtering systems. *Proc. IEEE/WIC COLA'03, Halifax, Canada,* 13–23.

Bhargava, H. K., D. Sun, S. H. Xu. 2006. Stockout compensation: Joint inventory and price optimization in electronic retailing. *INFORMS J. Comput.* **18** 255–266.

Brown, E. 2002. Slow road to fast data. *Fortune* (March 18), http://money.cnn.com/magazines/fortune/fortune_archive/2002/03/18/319878.

Chambers, L., ed. 1995. *Practical Handbook of Genetic Algorithms: Applications*, Vol. 1. CRC Press, Boca Raton, FL.

Cormen, T., C. Leiserson, R. Rivest. 2000. *Introduction to Algorithms*. MIT Press, Cambridge, MA.

Deo, N., P. Gupta. 2001. Sampling the web with random walks. *Congressus Numerantium, 149. 32nd Southeastern Internat. Conf. Combinatorics, Graph Theory and Comput.*, Baton Rouge, LA, 143–154.

Deo, N., P. Gupta. 2003. Graph-theoretic analysis of the World Wide Web: New directions and challenges. *Mathematica Contomporanea, Sociedade Brasileira de Matemática* **25** 49–69.

Drezner, Z. 2003. A new genetic algorithm for the quadratic assignment problem. *INFORMS J. Comput.* **15** 320–330.

Drezner, Z., S. Salhi. 2002. Using hybrid metaheuristics for the one-way and two-way network design problem. *Naval Res. Logist.* **49** 449–463.

Economist, The. 2002. E-commerce: Profits at last. *The Economist* (December 19) 95–96.

Garey, M. R., D. S. Johnson. 1979. *Computers and Intractability*. Freeman, New York.

Golub, G. H., C. F. Van Loan. 1997. *Matrix Computations*, 3rd ed. The Johns Hopkins University Press, Baltimore, MD.

Hoffman, W., J. Keedy, K. Roberts. 2002. The unexpected return of B2B. *The McKinsey Quarterly* 3, http://www.mckinseyquarterly.com.

Hong, W., J. Y. L. Thong, K. Y. Tam. 2004. Does animation attract online users' attention? The effects of flash on information search performance and perceptions. *Inform. Systems Res.* **15** 60–86.

Hurst, M. 1999. Building a great customer experience to develop brand, increase loyalty and grow revenues. White Paper One, Creative Good, Inc., http://www.creativegood.com.

Karuga, G. G., A. M. Khraban, S. K. Nair, D. O. Rice. 2001. AdPalette: An algorithm for customizing online advertisements on the fly. *Decision Support Systems* **32** 85–106.

Koopmans, T., M. Beckmann. 1957. Assignment problems and the location of economic activities. *Econometrica* **25** 53–76.

Lv, Q., P. Cao, E. Cohen, K. Li, S. Shenker. 2002. Search and replication in unstructured peer-to-peer networks. *Proc. ACM ICS'02*. 84–95.

Mathwick, C., N. Malhotra, E. Rigdon. 2001. Experiential value: Conceptualization, measurement and application in the catalog and internet shopping environment. *J. Retailing* **77** 39–56.

Michalewicz, Z. 1994. *Genetic Algorithms + Data Structures = Evolution Programs*, 2nd ed. Springer, New York.

Mullaney, T. J. 2004. E-biz strikes again! *Business Week* (May 10) 80–81.

Mullaney, T. J., H. Green, M. Arndt, R. D. Hof, L. Himelstein. 2003. The e-biz surprise. *Business Week* (May 12) 60–66.

Muralidhar, K., R. Sarathy, R. Parsa. 2001. An improved security requirement for data perturbation with implications for e-commerce. *Decision Sci.* **32** 683–698.

Nua Internet Surveys. 2003. How many online? *Nua Internet Surveys*, http://www.nua.com/surveys/how_many_online.

Norman, K. L. 1991. *The Psychology of Menu Selection*. Ablex, Norwood, NJ.

Padmanabhan, B., A. Tuzhilin. 2003. On the use of optimization for data mining: Theoretical interactions and eCRM opportunities. *Management Sci.* **49** 1327–1343.

Rosen, C. 2001. Amazon's alliances. *Information Week* (June 4), http://informationweek.com/840/amz_online.htm.

Sarwar, B., G. Karypis, J. Konstan, J. Riedl. 2001. Item based collaborative filtering recommendation algorithms. *Proc. WWW10 Conf.*, http://www10.org/cdrom/papers/contents.html.

Schaffer, E., J. Sorflaten. 2004. Key tips for user-centered design. *Human Factors International*, http://www.humanfactors.com/downloads/keytips.asp.

Schlosser, J. 2003. Looking for intelligence in ice cream: Companies have mastered collecting information, but not what to do with it. *Fortune* (March 17), http://money.cnn.com/magazines/fortune/fortune_archive/2003/03/17/339261.

Tan, Q. Z., X. Y. Chai, W. Ng, D. L. Lee. 2004. Applying co-training to clickthrough data for search engine adaptation. *Lecture Notes in Computer Science*, Vol. 2973. Springer, Berlin, Germany, 519–532.

Tan, Y., K. Moinzadeh, V. Mookerjee. 2005. Optimal processing policies for an e-commerce web server. *INFORMS J. Comput.* **17** 99–110.

Tedeschi, R. 2003. As the holidays approach, online retailers seek to spiff up their sites and minimize consumer frustation. *The New York Times* (October 27) C6–C7.