УДК 519

Оптимизация пользовательских меню с учётом семантического качества

М.В. Губко, А.И. Даниленко

При проектировании пользовательских интерфейсов часто возникает задача оптимизации иерархических меню. Существующие подходы к этой проблеме либо не учитывают смысловую нагрузку пунктов меню, либо не решают задачу оптимизации структуры меню. В статье предложена математическая модель оптимизации иерархических меню, учитывающая семантическое качество пунктов меню. На базе этой модели разработаны алгоритмы и реализован программный продукт для автоматической и автоматизированной оптимизации иерархических меню. Предложенный подход проиллюстрирован примером оптимизации голосового меню банка.

Ключевые слова: *оптимизация интерфейса, иерархическое меню, семантическое качество, автоматизация построения меню.*

Введение

Иерархические меню по-прежнему часто используются в пользовательских интерфейсах. Это командные меню офисного программного обеспечения, каталоги Интернет-сайтов, меню банкоматов, мобильных телефонов и современных бытовых приборов. Голосовые меню часто используется для реализации интерактивных телефонных сервисов и линий поддержки. Иерархическая организация позволяет пользователям быстрее найти искомую команду или раздел в системе.

В долгосрочной перспективе иерархические меню будут, по-видимому, использоваться всё реже, так как необходимость в них продиктована несовершенством современных технологий взаимодействия с пользователем. Если бы существующие технологии позволяли угадывать намерения пользователя (по крайней мере, в тех случаях, когда он точно знает, что именно он ищет), то не было бы необходимости в процессе пошагового уточнения цели пользователя, который, по сути, реализуется при навигации по панелям меню.

Технологии человеко-машинного взаимодействия развиваются крайне быстро, однако, так же быстро уменьшаются и размеры используемых устройств. Во многих популярных мобильных устройствах использование иерархических командных меню является

следствием ограниченности пользовательского интерфейса и средств ввода. Таким образом, разработка методов построения удобных меню, как и ранее, является актуальной задачей.

Помимо практической значимости, задача построения меню представляет интерес и с точки зрения применения формальных методов в теории человеко-машинного взаимодействия. Задача оптимизации пользовательских меню может служить образцом для демонстрации методов проектирования интерфейсов, основанных на математических моделях. Дело в том, что, с одной стороны, задача оптимизации меню чётко ограничена и достаточно легко формализуется, с другой стороны её решение, как будет показано ниже, требует разработки и апробации ряда взаимосвязанных аналитических, концептуальных и численных моделей (семантическая модель предметной области, модели различных типов меню, психологические модели поведения пользователей в меню, математическая модель оптимизации меню и так далее). Результатом комбинации этих моделей и методов является описываемая в статье прикладная методика оптимизации иерархических меню. Предлагается гибкий способ учёта семантических аспектов при оптимизации иерархической структуры меню. Результаты реализованы в интерактивном инструменте поддержки разработки структуры пользовательских меню, эффективность подхода проверена на примере оптимизации нескольких реальных меню.

1. Обзор литературы

Исследования иерархических пользовательских меню начались с момента появления первых графических интерфейсов. В наши дни иерархические структуры часто исследуются в применении к интерфейсам с ограниченными возможностями, таким как бытовые приборы [1] или голосовые меню, и для описания структуры Интернет-сайтов [2–4].

Наиболее исследованный (и наиболее легкоформализуемый) аспект разработки меню – влияние ширины и глубины меню на его привлекательность и удобство для пользователя. Что лучше – широкое и неглубокое меню, в котором пользователю одновременно предлагается больше альтернатив, но при этом требуется меньше шагов для достижения цели? Или лучше предложить более узкую и глубокую структуру, которая предлагает всего несколько вариантов на каждом уровне, но имеет большее количество уровней? Ответ на этот вопрос оказывается неоднозначным.

Существуют различные подходы к этой проблеме. Экспериментальные исследования говорят в пользу иерархических структур, содержащих примерно 8 вариантов в панели меню [5, 6]. Большинство аналитических исследований [7, 8] подтверждают, что оптимальное число вариантов находится в пределах от 4 до 13. Точные значения варьируются в

зависимости от методов исследования и рассматриваемых условий. В [7] предлагается «линейная» модель навигации в меню, из которой следует, что в «хорошей» панели меню должно быть от 6 до 8 вариантов. В [8] утверждается, что в случае, когда варианты в панели меню отсортированы, т.н. «логарифмическая модель» лучше описывает поведение пользователей. Авторы исследования [9] комбинируют закон Хика-Хаймана и закон Фиттса для описания поведения пользователя и вычисления оптимальной структуры меню.

Но в реальности знания оптимальной ширины меню недостаточно для построения качественного меню. Второй задачей является наполнение структуры меню осмысленными вариантами, которые будут понятными пользователю. Важность учёта семантических аспектов при построении пользовательских интерфейсов подчёркивается уже на протяжении долгого времени (как минимум, со времён издания книги [10]), но эти соображения находят отражение далеко не во всех исследуемых моделях.

На сегодняшний день существует несколько имитационных моделей, учитывающих семантические аспекты вариантов в меню. Модель MESA (Method for Evaluating Site Architecture) [4] имитирует стратегию поведения пользователей, основанную на значениях релевантности, назначенных для ярлыков категорий в меню. К сожалению, авторы не обсуждают методы определения этих значений. В моделях CoLiDeS (Comprehension-based Linked model of Deliberate Search) [3] и SNIF-ACT (Scent-based Navigation and Information Foraging in the ACT architecture) [2] релевантность ярлыков категорий определяется исходя из сходства с заголовком искомой функции.

Существуют также исследования [11] посвящённые вопросу применения латентно-семантического анализа (ЛСА) [12] к определению качества ярлыков элементов в пользовательском интерфейсе. Позже [2] ЛСА был применён для исследования структуры Интернет-сайтов. Главное ограничение при применении ЛСА для проектирования пользовательских интерфейсов – это необходимость в значительном объёме данных. В [13] показано, что для получения качественных результатов каждый элемент должен характеризоваться 100–200 словами.

Альтернативный подход заключается в использовании простых методов, таких как «сортировка карточек» (card sorting) или статистический анализ содержимого [14]. Было показано [15], что подобные подходы позволяют получить классификации функций приемлемого качества.

Описанные методы позволяют получить осмысленную классификацию функций, которая может быть использована при проектировании структуры меню. Однако эти подходы не решают задачу поиска оптимальной структуры.

Одна из известных авторам оптимизационных моделей, которая учитывает семантические аспекты, описана в [16]. Работа развивает подход, предложенный в [7], снимая часть ограничений, и предлагает алгоритм построения оптимальной структуры меню с семантическими ограничениями. Алгоритм использует иерархические классификации функций для формирования осмысленных группировок. В результате поиск оптимальной структуры меню происходит только среди структур, полученных удалением промежуточных вершин из первоначальной подробной классификации.

Ранее авторами была предложена математическая модель оптимизации меню [17, 18], основанная на минимизации среднего времени поиска в иерархической структуре меню. Аналитические результаты, дополненные иерархическими классификациями функций, позволяют построить автоматизированную процедуру итеративного улучшения структуры меню [18]. В отличие от подхода, предложенного в [16], используется одновременно несколько классификаций функций для формирования осмысленных группировок. Например, функции в меню могут быть сгруппированы по типу действия (просмотр, правка, удаление) или по объекту действия (файл, сообщение, абзац). Алгоритм выбирает оптимальный срез в дереве классификации для формирования панели меню.

Основным недостатком этих подходов является то, что при оптимизации используются семантические ограничения, а не семантическое качество ярлыков. Другими словами, не учитывается зависимость времени поиска в панели меню от смысла представленных в ней категорий; предполагается, что время восприятия любого из допустимых вариантов одинаково.

В настоящей статье описывается оптимизационная модель, в которой принимается во внимание семантическое качество пунктов меню. Также предлагается полностью автоматический алгоритм построения осмысленного меню с приближённо минимальным средним временем доступа к функциям. Полученные результаты реализованы в интерактивном инструменте оптимизации иерархических меню.

2. Математическая модель оптимизации меню

Задача любого меню – обеспечить быстрый доступ к набору функций путём организации их в иерархию категорий. При поиске функции пользователь проводит всё время, просматривая и совершая действия в различных панелях меню. Стратегия, применяемая пользователем для поиска варианта в меню, зависит от ряда факторов, таких как тип меню и квалификация пользователя. В [5] приводится подробное обсуждение различных стратегий пользователя. Когда пользователь точно представляет, что он ищет,

ему остаётся только сопоставить цель с вариантами в панелях меню. В этом случае, как правило, пользователь придерживается стратегии последовательного поиска, то есть просматривает варианты в панели меню последовательно, пока не дойдёт до подходящей категории. В иных случаях возможно применение исчерпывающего поиска (пользователь просматривает все варианты перед осуществлением выбора) или случайного поиска (пользователь просматривает варианты в случайном порядке). Вне зависимости от применяемой стратегии б**о**льшая часть времени тратится на просмотр и анализ вариантов, которые пользователю не нужны.

При этом ярлыки, соответствующие просматриваемым вариантам, очевидно, имеют различное качество. Короткие и ёмкие названия воспринимаются легко, в то время как длинные и неоднозначные требуют гораздо больше внимания. Подобные свойства ярлыков будем называть их семантическим качеством. В зависимости от типа пользовательского интерфейса, частью которого является меню, роль ярлыка играет текстовое название, голосовая метка или пиктограмма. Семантическое качество имеет различный смысл для каждого из этих случаев. Например, при навигации неопытного пользователя в голосовом меню время, необходимое для выбора i-го варианта в панели меню, складывается из времени воспроизведения голосовых меток от первой до i-й.

Длина ярлыка (длина текстового названия или время воспроизведения голосовой метки) является самым простым примером семантического качества, которым мы будем пользоваться ниже для иллюстрации. В общем случае семантическое качество описывает не только время чтения ярлыка (длина текста), но и время анализа (понятность текста), время совершения выбора (удобство расположения ярлыка), читаемость пиктограмм или размер кнопок (согласно закону Фиттса).

Кроме того, семантическое качество не ограничивается оценкой времени чтения или анализа ярлыков. Другим аспектом является вероятность ошибки пользователя. Двусмысленный ярлык категории приводят к тому, что пользователь будет чаще выбирать эту категорию по ошибке. В заключительном разделе показано, как предлагаемая модель может быть расширена для учёта такого способа оценки семантического качества.

Сформулируем задачу построения структуры меню в терминах теории оптимизации иерархических структур.

Рассмотрим множество функций $N = \{1, \ldots, n\}$, которые требуется разместить в меню. Каждой функции $w \in N$ поставим в соответствие её популярность $\mu(w)$. Будем считать, что целью пользователя в тот или иной момент времени является только одна функция. Иерархическое меню описывается деревом, листьями которого являются функции из

множества N, а промежуточные вершины соответствуют категориям. Каждая категория при этом характеризуется множеством составляющих её функций $s \subseteq N$.

Не все группировки функций являются допустимыми. Будем считать, что задано некоторое количество классификаций функций системы по различным основаниям (например, функции офисного приложения можно группировать по типу действия – просмотр, правка, удаление – или по объекту действия – файл, сообщение, абзац). Каждая классификация задаёт своё разбиение множества функций N на непересекающиеся подмножества – категории. При построении панели меню можно использовать только категории, задаваемые одной из заданных классификаций, при этом запрещено комбинировать категории разных классификаций во избежание неравноположных перечислений. В то же время, сами классификации могут быть иерархическими, и панель меню может «собираться» из категорий разных уровней (например, можно использовать или категорию «Ссылки», или составляющие её подкатегории «Оглавление», «Сноски», «Список литературы», «Названия» и «Предметный указатель»).

Каждой функции или допустимой категории (множеству функций) соответствует некоторый ярлык (описывающий функцию или категорию одной из имеющихся классификаций). Поставим в соответствие каждому ярлыку l его семантическое качество $\omega(l)$. Для большей наглядности алгоритмы ниже описаны для случая голосового меню, в котором семантическое качество $\omega(l)$ определяется временем воспроизведения голосовой метки l.

Рассмотрим категорию $s \subseteq N$, состоящую из k подкатегорий $s_1, \ldots, s_k$ (подкатегория может состоять и из единственной функции). Предположим, что категории $s_1, \ldots, s_k$ формируют панель меню с ярлыками $l_1, \ldots, l_k$ соответственно. Обозначим для краткости $\omega_i := \omega(l_i)$. Тогда время, требуемое для поиска i-го варианта в панели меню, с учётом семантического качества ярлыков можно записать как $t_i(k, \omega_1, \ldots, \omega_k)$. На практике влияние ширины меню k и семантических аспектов $\omega_1, \ldots, \omega_k$ можно разделить. В этом случае время на поиск варианта в панели меню можно представить в виде $t_i(k,\omega_1,\ldots,\omega_k) = \hat{t}_i(k) \times \sigma_i(\omega_1,\ldots,\omega_k)$, где функция $\hat{t}_i(k)$ характеризует влияние структуры меню, а функция $\sigma_i(\omega_1, \ldots, \omega_k)$ описывает влияние семантики. Например, для голосового меню в качестве семантического качества разумно рассматривать время воспроизведения метки. Если пользователи при этом применяют стратегию последовательного поиска, то время на выбор i-го варианта в панели меню определяется как $t_i(k, \Omega) = \omega_1 + \ldots + \omega_i + t_0$, где t_0 – это время, требуемое на совершение выбора (нажатие на клавишу телефона).

В общем случае время, затрачиваемое пользователем в панели меню, определяется позицией искомого варианта, общим количеством вариантов в панели (шириной меню) и семантическим качеством всех вариантов (не только искомого варианта). Все остальные параметры (такие как тип меню, стратегия пользователя, возможности оборудования) отражаются в виде функций $\hat{t}_i(k)$ и $\sigma_i(\omega_1, \ldots, \omega_k)$. Такой подход позволяет описывать разнообразные типы меню и стратегии пользователей. Аналогичный подход, но без функции семантики $\sigma_i(\omega_1, \ldots, \omega_k)$, был использован в [17, 18] для описания стратегий последовательного поиска, исчерпывающего поиска, а также учёта вероятности ошибки пользователя.

Обозначим через y_i относительную популярность i-го пункта в панели меню, то есть частоту, при которой пользователю требуется именно этот пункт, в случае, если пользователь вошёл в данную панель. Тогда среднее время, которое пользователь проводит в панели меню, вычисляется как взвешенная сумма времени выбора каждой из альтернатив:

$$(1) \qquad t(y_1,\ldots,y_k,\omega_1,\ldots,\omega_k) = \sum_{i=1}^{k} y_i \hat{t}_i(k)\sigma_i(\omega_1,\ldots,\omega_k), \sum_{i=1}^{k} y_i = 1.$$

Чем больше значение функций $\sigma_i(\cdot)$, тем больше время, проведённое в панели меню, что соответствует худшему семантическому качеству используемых ярлыков.

Например, для стратегии последовательного поиска (при которой пользователь последовательно знакомится с пунктами меню до тех пор, пока не достигает нужного), среднее время пребывания пользователя в меню определяется выражением

$$(2) \qquad t(y_1,\ldots,y_k,\Omega) = t_0 + \sum_{i=1}^{k} y_i \sum_{j=1}^{i} \omega_j .$$

Всё время, проводимое в иерархическом меню, пользователь тратит на выбор вариантов в различных панелях меню. Таким образом, среднее время поиска в меню H вычисляется как сумма времён, затрачиваемых в каждой панели меню s, помноженных на популярность соответствующих этим панелям категорий (то есть на вероятность того, что пользователь зайдёт в эту панель меню)

$$T(H) = \sum_{s \in H} \mu_s \cdot t\left(\mu_{s_1}/\mu_s, \ldots, \mu_{s_{k(s)}}/\mu_s, \omega_1, \ldots, \omega_{k(s)}\right),$$

где $k(s)$ – количество вариантов в панели меню s, $\mu_{s_1},\ldots,\mu_{s_{k(s)}}$ – абсолютные популярности вариантов, а $\omega_1, \ldots, \omega_{k(s)}$ – семантическое качество их ярлыков.

Задача поиска оптимальной структуры меню состоит в том, чтобы для заданного множества функций N найти структуру меню H, состоящую из осмысленных категорий, для которой среднее время поиска $T(H)$ минимально.

В такой постановке задача является расширением базовой модели, описанной в [17, 18], за счёт учёта семантического качества. При этом базовая модель является частным случаем, когда семантическое качество всех ярлыков в меню одинаково.

Если все ярлыки в меню имеют одинаковое семантическое качество $\hat{\omega}$, то можно использовать нижнюю оценку среднего времени поиска в меню, полученную в [17, 18]:

$$T_L\left(N,\hat{\omega}\right)=\left(\mu_N\ln\mu_N-\sum_{w\in N}\mu(w)\ln\mu(w)\right)\cdot\min_k\min_{y_1,\dots,y_k}\frac{t\left(y_1,\dots,y_k,\hat{\omega},\dots,\hat{\omega}\right)}{-\sum_{i=1}^{k}y_i\ln y_i}. \quad (3)$$

Минимизация проводится по всем возможным значениям ширины меню $k = 2, 3, \dots$ и по всем пропорциям – комбинациям вектора относительных популярностей $y_1, \dots, y_k$.

Пусть минимум в выражении (3) достигается при ширине меню r и пропорции $x_1, \dots, x_r$. Это решение задачи минимизации тесно связано с видом оптимальной структуры меню. Каждая панель в оптимальном меню в идеале должна содержать ровно r вариантов с относительными популярностями $x_1, \dots, x_r$. В [18] предложен итерационный алгоритм построения осмысленной структуры меню, в котором иерархия строится сверху вниз, при этом наполнение панелей меню подбирается на основе предоставленных классификаций функций путём подстановки их в оптимальный шаблон $x_1, \dots, x_r$.

Но, как было упомянуто выше, нижняя оценка (3) не учитывает семантические аспекты. Можно вычислить нижнюю оценку среднего времени поиска в меню, приняв наилучшее семантическое качество для всех ярлыков. Но качество этой оценки будет значительно хуже, и на неё нельзя полагаться в алгоритме оптимизации. Вместо этого в описываемых ниже алгоритмах используется некоторое среднее значение семантического качества $\hat{\omega}$ в выражении (3) для вычисления оценки среднего времени поиска в меню.

3. Алгоритм оптимизации меню

Когда семантические аспекты не учитываются, нижняя оценка (3) позволяет описать оптимальную структуру меню. Алгоритм, предложенный в [18] для формирования панели меню, группирует функции в набор категорий, наилучшим образом подходящий под оптимальный шаблон.

Для рассматриваемой модели, учитывающей семантические аспекты, оптимальная структура каждой панели меню зависит от семантического качества ярлыков. Это не позволяет применить предложенный в [18] подход. Вместо использования шаблона для построения меню предлагается жадный алгоритм оптимизации меню сверху вниз.

Идея любого жадного алгоритма состоит в сведении глобальной задачи оптимизации к множеству независимых локальных задач, решение которых даёт результат, близкий к глобальному оптимуму. В терминах задачи оптимизации меню локальными задачами является построение каждой отдельной панели меню, а именно, для панели меню s необходимо определить количество вариантов k, разбив множество элементов s на допустимые подкатегории $s_1, \ldots, s_k$, для всех подкатегорий определить ярлыки $l_1, \ldots, l_k$ и определить порядок их следования в панели меню. В идеале построение локально оптимальных панелей меню даст в результате глобально оптимальную структуру меню.

Построение меню начинается с верхнего уровня и повторяется для каждой подкатегории вплоть до нижнего уровня. Структура каждой панели меню выбирается путём перебора всех доступных классификаций, в случае иерархической классификации конкретное разбиение функций системы на категории выбирается локальным поиском по этой иерархической классификации. В результате выбирается панель меню, минимизирующая локальный критерий оптимизации – оценку времени поиска в результирующем меню. Оценка вычисляется как сумма времени поиска в текущей панели меню и аналитически вычисленной оценки времени поиска в порождаемых ею иерархических подменю, т.е. время поиска в текущей панели меню вычисляется по формуле (1) и суммируется с оценкой времени для каждого подменю, вычисленной по формуле (3). Алгоритм выполняется несколько раз для различных значений «среднего» семантического качества $\hat{\omega}$, и выбирается результат с наилучшими показателями.

Для реализации алгоритма необходимо решить несколько задач. Во-первых, необходим метод поиска различных разбиений функций панели меню на непересекающиеся осмысленные категории. Кроме того, необходимо знать, каким образом сортировать категории в панели меню. И в третьих – должна быть возможность сравнивать различные варианты наполнения панели меню, т.е. необходимо определить критерий оптимальности.

3.1. Поиск разбиения функций

Для поиска осмысленного разбиения функций на категории алгоритм использует иерархические классификации функций, предоставленные разработчиком меню. Пример возможной классификации функций банковского меню приведён на рис. 1 (категории выделены жирным шрифтом). Из каждой классификации можно составить множество осмысленных непересекающихся разбиений функций. Например, на основании классификации, приведённой на рис. 1, можно разбить функции на категории «Информация по счетам», «Осуществление банковских переводов», «Настройка финансовых свойств» и

«Настройка технических параметров». Другим вариантом разбиения является набор категорий «Информация о состоянии счёта и операциях по счёту», «Выписки по вашим счетам», «Получение банковских форм и заявлений», «Осуществление банковских переводов», «Настройка» и отдельно стоящей функции «Размещение срочных депозитов». Оба эти разбиения покрывают полный набор функций.

Информация по счетам
- **Информация о состоянии счёта и операциях по счёту**
  - Узнать остатки по вашим банковским счетам
  - Узнать доступный лимит по кредитной карте
- **Выписки по вашим счетам**
  - Заказ выписки по кредитной карте
  - Получение выписки по счёту
- **Получение банковских форм и заявлений**
  - Получение банковских заявлений
  - Получение одной из банковских форм
- Размещение срочных депозитов

Осуществление банковских переводов
- Осуществление переводов между вашими счетами
- Погашение задолженности по кредитной карте
- Осуществление коммунальных платежей
- Осуществление переводов третьим лицам
- Осуществление перевода по программе «Заплати в рассрочку»

Настройка
- **Настройка финансовых свойств**
  - Активация получателя
  - Увеличение лимита по кредитной карте
- **Настройка технических параметров**
  - Изменение ТПИНа или ПИН-кода
  - Изменение основного текущего счёта по карте
  - Создание собственного меню в системе

Рис. 1. Пример классификации функций

Для каждой доступной классификации алгоритм использует локальный поиск для выбора оптимального разбиения. Ниже поясняется, каким образом сравниваются различные разбиения и как выбирается наилучшее.

Локальный поиск обходит иерархию классификации, формируя различные разбиения. Результат определяются направлением поиска и отправной точкой. Было опробовано шесть вариантов локального поиска:

1. Поиск сверху вниз с набором категорий верхнего уровня в качестве отправной точки. На каждом шаге алгоритм пробует улучшить качество разбиения, заменяя каждую из категорий её содержимым в дереве классификации, таким образом, двигаясь от корня дерева в направлении детализации категорий. Для примера, приведённого на рис. 1, алгоритм

стартует с набора категорий «Информация по счетам», «Осуществление банковских переводов» и «Настройка». На следующем шаге категорий «Информация по счетам» будет заменена на её содержимое, т.е. на категории «Информация о состоянии счёта и операциях по счёту», «Выписки по вашим счетам», «Получение банковских форм и заявлений» и отдельно стоящую функцию «Размещение срочных депозитов». Далее аналогичное действие будет проделано с категорией «Осуществление банковских переводов» и т.д.

2. «Жадный» поиск сверху вниз действует аналогично описанному выше алгоритму, но разворачивает только самую популярную категорию. В качестве отправной точки также используется набор категорий верхнего уровня.

3. Поиск сверху вниз в качестве отправной точки использует набор отдельных функций. На каждом этапе алгоритм пробует улучшить качество разбиения, сворачивая несколько элементов в категорию, который они принадлежат в классификации. Таким образом, поиск происходит от самого широкого разбиения вверх к корню дерева.

4. Двунаправленный поиск на каждом шаге пробует как развернуть каждую из категорий, так и свернуть элементы разбиения в общие категории. В качестве отправной точки используется набор категорий верхнего уровня.

5. Вариант двунаправленного поиска, использующий полный набор функций в качестве отправной точки.

6. Двунаправленный поиск, использующий в качестве отправной точки субоптимальное разбиение, которое построено по шаблону, рассчитанному при помощи минимизации (3) без учёта семантических аспектов.

Описанные варианты локального поиска были опробованы на ряде типовых классификаций. Оказалось, что все варианты двунаправленного поиска в результате дают тот же результат, что либо поиск сверху вниз, либо поиск снизу вверх. Таким образом, их можно исключить без потери качества алгоритма оптимизации. Дальнейшие эксперименты показали, что жадный поиск сверху вниз в подавляющем большинстве случаев приводит к тому же результату, что полный поиск сверху вниз. В случае различных результатов отклонения незначительны, и ими можно пренебречь ради существенного увеличения скорости работы алгоритма за счёт исключения поиска сверху вниз.

3.2. Сортировка вариантов в панели меню

Алгоритмы, описанные в предыдущем разделе, работают в том случае, когда классификации определяют не только допустимые группировки функций, но и их порядок следования в меню. Если разработчик меню может менять порядок функций, то

дополнительно возникает задача поиска оптимального упорядочения вариантов в панели меню.

Полный перебор всех вариантов сортировки в панели меню, содержащей k вариантов, требует рассмотрения $k!$ перестановок, что неприемлемо с точки зрения скорости работы алгоритма. С другой стороны, когда все варианты в панели меню имеют одинаковое семантическое качество, и единственной их характеристикой является относительная популярность y_i, существует простое решение – чем выше популярность элемента y_i, тем меньше времени должно требоваться для доступа к этому элементу.

Задачу сортировки с учётом семантического качества вариантов можно решить также для конкретных стратегий поведения пользователя.

Например, рассмотрим панель меню, содержащую k вариантов. Каждому варианту $i = 1, \ldots, k$ соответствует его относительная популярность y_i и время чтения текстовой метки ω_i. Если пользователь придерживается стратегии последовательного поиска (что типично, в частности, для голосовых меню), то среднее время поиска в этой панели меню определяется выражением (2). Необходимо минимизировать это выражение перестановкой вариантов в панели меню, что является классической задачей теории расписаний. Согласно правилу МакНотона [19] решением является сортировка вариантов по возрастанию величины ω_i/y_i.

Пусть пользователь придерживается стратегии случайного поиска – варианты в панели меню просматриваются в случайном порядке, пока не будет найден искомый. В этом случае порядок следования вариантов в меню неважен, т.к. при любом порядке среднее время поиска будет одинаковым.

Аналогичные решения существуют для большинства типов меню и стратегий поведения пользователей. В то же время, в общем случае не существует простого решения для описания наилучшей сортировки вариантов.

3.3. Локальный критерий оптимизации

Структура каждой панели меню выбирается путём минимизации оценки времени поиска в результирующем подменю с корнем в текущей панели. Рассмотрим панель меню $s \subseteq N$, которая содержит k подкатегорий $s_1, \ldots, s_k$ с относительными популярностями $y_1, \ldots, y_k$ и семантическим качеством $\omega_1, \ldots, \omega_k$.

В общем случае время поиска в подменю зависит от структуры меню вплоть до терминальных вершин (функций). Но для локализации вычислений можно использовать оценку (3) для определения времени, затрачиваемого в подкатегориях (структуру которых мы не знаем на этапе построения текущей панели меню). Таким образом, среднее время

$\hat{T}(s, s_1, ..., s_k, \omega_1, ..., \omega_k, \hat{\omega})$ поиска в подменю с корнем в категории *s* зависит от набора вариантов $s_1, \ldots, s_k$, представленных в панели меню *s*, от семантического качества их ярлыков $\omega_1, \ldots, \omega_k$ и от «среднего» семантического качества $\hat{\omega}$, принятого для всех ярлыков в дочерних панелях меню.

Пусть все ярлыки в каждой из *k* дочерних панелей меню имеют одинаковое семантическое качество $\hat{\omega}$. При таком «усреднении» семантических свойств можно использовать выражение (3) для вычисления среднего времени поиска в любом подменю $T_L(s_i, \hat{\omega})$. Для текущей панели меню *s* можно вычислить точное время поиска по формуле (1). Таким образом, оценка времени поиска в подменю с корнем в категории *s*, взвешенная на популярность этой категории, определяется выражением

$$\begin{aligned} \hat{T}\left(s, s_1, ..., s_k, \omega_1, ..., \omega_k, \hat{\omega}\right) &= \mu_s t\left(\mu_{s_i}/\mu_s, ..., \mu_{s_k}/\mu_s, \omega_1, ..., \omega_k\right) + \\ &+ \sum_{i=1}^{k}\left(\mu_{s_i}\ln\mu_{s_i} - \sum_{w\in s_i}\mu(w)\ln\mu(w)\right)\cdot \min_r \min_{x_1,...,x_r} \frac{t\left(x_1, ..., x_r, \hat{\omega}, ..., \hat{\omega}\right)}{-\sum_{j=1}^{r} x_j \ln x_j}. \end{aligned} \tag{4}$$

Критерий $\hat{T}(\cdot, \hat{\omega})$, вычисляемый согласно формуле (4), используется алгоритмом для выбора структуры каждой панели меню.

3.4. Шаги алгоритма

В итоге алгоритм автоматической оптимизации иерархического меню описывается следующей последовательностью шагов.

1. Зафиксировать начальное значение «среднего» семантического качества $\hat{\omega}$ равным наилучшему из всех меток семантическому качеству.

2. Для каждой доступной классификации локальным поиском снизу вверх и «жадным» локальным поиском сверху вниз найти наилучшее (в терминах описанного критерия) разбиение функций на осмысленные категории. Выбрать наилучшее разбиение среди различных классификаций и использовать его для формирования верхней панели меню.

3. Для каждой сформированной категории повторить процесс наполнения соответствующих панелей меню вплоть до терминальных вершин.

4. Повторить шаги 2 и 3 для всех $\hat{\omega}$ с заданным интервалом вплоть до значения, соответствующего наихудшему из всех меток семантическому качеству.

5. Среди найденных структур меню выбрать структуру с наименьшим средним временем поиска.

3.5. Оценка качества алгоритма

Предлагаемый алгоритм использует различные эвристики и локальный поиск для достижения приемлемой скорости работы. Это могло повлиять на качество результирующего меню. К счастью, математическая модель предоставляет методы оценки качества результирующей структуры.

Если за $\hat{\omega}$ взять наилучшее значение семантического качества, то выражение (3) даст нижнюю оценку среднего времени поиска в оптимальном меню с учётом семантического качества. Эту нижнюю оценку можно использовать для определения качества структур, полученных при помощи предлагаемого алгоритма.

Дополнительная минимизация по семантическому параметру $\hat{\omega}$, очевидно, отрицательно влияет на качество этой нижней оценки. Меню, построенные автоматически с учётом семантического качества, показывают отклонение в 30% от теоретического минимума. Однако более детальный анализ показывает, что значительная часть этого расхождения является результатом ухудшения оценки из-за семантических аспектов, и нет практической возможности построить меню, которое было бы заметно лучше. Исследования на ряде типовых меню показывают, что среднее время поиска в меню, сформированных автоматически, находится в пределах 10% от среднего времени поиска для наилучшего меню, которое может быть построено на практике.

4. Инструмент оптимизации меню

Описанный выше алгоритм реализован в инструменте автоматической оптимизации иерархических меню. Инструмент также предоставляет разработчику меню средства для ручного и полуавтоматического редактирования структуры (например, для внесения неформальных ограничений в результирующую структуру меню).

В первую очередь разработчик меню должен загрузить исходные данные – набор функций меню с их популярностями и несколько (желательно, иерархических) классификаций этих функций. Далее разработчику предлагается выбрать окружение, в котором будет использоваться меню – тип меню (например, выпадающий список, таблица пиктограмм и т.д.), стратегию навигации в меню и другие характеристики. Реализованный инструмент содержит несколько предустановленных вариантов для типичных сценариев, а также предоставляет возможность загрузить экспериментальные данные для специфических случаев. Как только настройка окружения завершена разработчику меню предлагаются средства оптимизации.

На рис. 2 изображён интерфейс разработанного инструмента. Интерфейс разделён на две части – в левой части содержатся исходные данные и аналитические вычисления, а правая часть служит для работы над результирующим меню.

В левой части приложения доступна таблица с исходными функциями, также имеется возможность просмотреть их в структуре любой из классификаций. Если задача состоит в улучшении существующего меню, исходная структура также может быть загружена в инструмент оптимизации. В этом случае интерфейс предоставляет удобные средства сравнения исходной и результирующей структур – этот режим изображён на рис. 2.

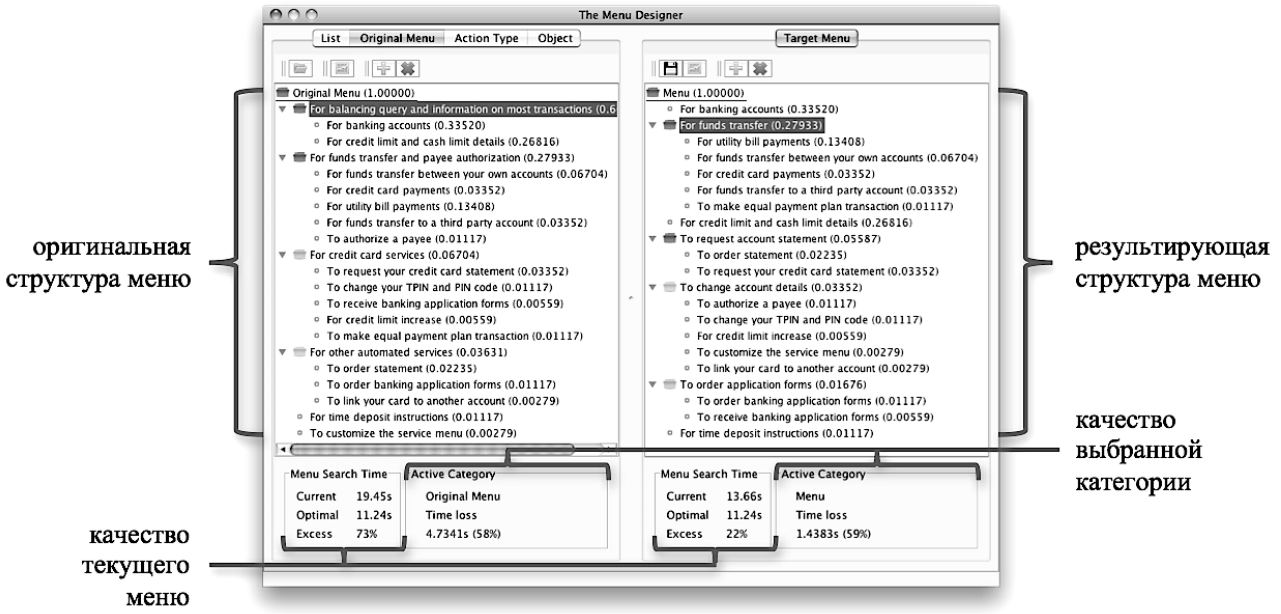

Рис. 2. Интерфейс инструмента оптимизации меню

Правая часть интерфейса содержит средства оптимизации структуры меню. В первую очередь приложение реализует описанный выше автоматический алгоритм оптимизации меню. Другими словами, инструмент позволяет нажатием одной кнопки получить осмысленную структуру меню хорошего качества в терминах среднего времени доступа. Если при этом есть необходимость учесть какие-либо неформальные ограничения в меню, разработчику меню доступны инструменты ручной и полуавтоматической корректировки структуры. Разработчик может изменять порядок следования вариантов в панели меню, перемещать функции в другую панель меню, добавлять и удалять категории, автоматически сортировать варианты в панели меню и оптимизировать структуру в отдельной части меню или в меню целиком. В процессе изменения структуры меню инструмент помогает в принятии решений, предоставляя аналитические оценки качества как отдельных панелей меню, так и меню в целом. Качество категорий визуально отображается цветом

соответствующих иконок в иерархии (зеленый, желтый или красный), что позволяет быстро найти категории, на которые следует обратить внимание в первую очередь. Также под каждой иерархией отображается численная оценка качества текущего меню и выделенной категории.

После того как формирование меню завершено, разработчик может сохранить результат в файл формата Microsoft® Excel® и реализовать сформированную структуру в реальной системе.

Инструмент оптимизации меню доступен для загрузки с сайта проекта http://www.mtas.ru/person/goubko/themenudesigner/. Там же можно найти более подробные инструкции по работе с программой и ссылки на её исходный код.

5. Пример оптимизации меню

Наиболее очевидным приложением модели оптимизации меню с учётом семантического качества являются голосовые меню (интерактивные телефонные системы). Ниже мы рассмотрим простой пример использования реализованного инструмента для улучшения голосового меню международного банка.

Оригинальная структура меню изображена в левой части интерфейса на рис. 2, она содержит 17 функций. При помощи статистических исследований и экспертных мнений была проведена оценка популярностей функций и категоризация их по двум основаниям – по типу и по объекту действия. Рабочее голосовое меню использовалось для замеров времени воспроизведения названий вариантов и других временны**х** параметров оборудования. В результате численного анализа было получено, что время воспроизведения метки длины L составляет $(1.3 + 0.065 \cdot L)$ секунд. Также замеры показали, что системе требуется одна секунда на переключение текущего меню, время совершения действия пользователем в среднем составляет также одну секунду. Используя эти значения, вычислено среднее время доступа к функциям оригинального меню – 19,45 сек.

В правой части приложения на рис. 2 отображена автоматически сформированная структура меню. Среднее время доступа к функциям в новом меню составляет 13,66 сек, то есть без вмешательства разработчика было получено улучшение в 30% относительно оригинальной структуры меню.

Оценим эффект от введения семантических аспектов в модель. При оптимизации рассматриваемой структуры меню при помощи модели, не учитывающей семантическое качество, результатом является другая структура со средним временем доступа 14,31 сек. Таким образом, на данном примере учёт семантики позволяет сэкономить дополнительные

0,65 сек. (5%) на каждую пользовательскую сессию. Принимая во внимание крайне малый размер иллюстративного примера (всего 17 функций), этот выигрыш является существенным.

6. Обсуждение и перспективы

Предложенная модель принимает во внимание семантические аспекты ярлыков в меню, такие как время воспроизведения в голосовых меню или чтения в программных меню. При описанном подходе семантическое качество ω может быть вектором, компоненты которого описывают различные семантические характеристики. Таким образом, модель позволяет достаточно гибко комбинировать несколько компонент семантического качества. Как следует из формулы (2), семантическое качество ярлыка в панели меню влияет на время поиска пользователем этого варианта, а также и других вариантов в этой панели меню.

Но формула (2) не учитывает другой важный аспект семантического качества – вероятность ошибки пользователя. Пользователь чаще по ошибке выбирает варианты с неясными или двусмысленными ярлыками. А совершение ошибки приводит к дополнительным затратам времени.

Для учёта такого рода временных затрат в первую очередь необходимо иметь модель навигации с вероятностью ошибки. Такая модель описана в [17, 18]. Согласно предложенному подходу, в формулу (2) необходимо добавить дополнительный член $p \cdot t_e(k)$, характеризующий временные затраты из-за ошибок пользователя. Здесь $t_e(k)$ – время, которое пользователь потратит в панели меню, если он попал в неё по ошибке, p – вероятность совершения ошибки в панели меню. Если вектор семантического качества ω включает компоненту, характеризующую ясность ярлыка, то вероятность по ошибке попасть в панели меню, содержащую k вариантов с ярлыками, имеющими семантическое качество $\omega_1, \ldots, \omega_k$, зависит не только от набора $\omega_1, \ldots, \omega_k$, но и от семантического качества ω' ярлыка, описывающего текущую категорию в панели меню на уровень выше. Таким образом, дополнительный член в (2), характеризующий вероятность ошибки и ясность ярлыков, принимает вид $p(\omega_1, \ldots, \omega_k, \omega') \cdot t_e(k)$.

Изменения в выражении для среднего времени поиска в панели меню приводят к изменениям в локальном критерии оптимизации (4), используемом в алгоритме оптимизации меню. Оценка времени поиска в подменю, используемая в выражении (4), складывается из точного времени поиска в текущей панели меню и оценок времени поиска в подкатегориях, вычисленных исходя из некоторого среднего значения семантического качества $\hat{\omega}$. Если теперь вероятность ошибки пользователя зависит от семантического качества ярлыка, время

поиска в подкатегориях зависит от семантического качества соответствующих ярлыков в текущей панели меню. Таким образом, критерий оптимизации (4) видоизменяется для расчёта трёх компонент времени поиска – во-первых, времени нахождения в текущей панели меню, во вторых – в панелях меню следующего уровня и в третьих – во всех нижележащих панелях меню.

Время поиска в текущей панели меню рассчитывается исходя из семантического качества ярлыков в ней, я вероятность попадания в эту панель по ошибке зависит от семантического качества соответствующего ярлыка в панели меню уровнем выше, которая зафиксирована на предыдущем шаге алгоритма. Оценка времени поиска в панелях меню следующего уровня вычисляется исходя из значения среднего семантического качества $\hat{\omega}$ для ярлыков этих панелей меню, но для вычисления вероятности попадания в эти панели меню по ошибке используются реальные значения семантического качества соответствующих ярлыков в текущей панели меню. Оценка для всех последующих уровней вычисляется, как и раньше, по формуле (3), в которой принимается среднее семантическое качество $\hat{\omega}$ ярлыков как для расчёта времени поиска, так и вероятности ошибки.

Для фиксированного разбиения вычисление модифицированного критерия сводится к решению k оптимизационных задач (для вычисления оптимальной ширины панелей меню следующего уровня). Вычислительная сложность этих задач не зависит от размеров меню и приемлема для осуществления вычислений в реальном времени.

Другим возможным развитием модели является работа с меню, имеющими недревовидную структуру, которая позволяет описывать альтернативные пути к функциям меню. Реальные интерфейсы, как правило, предоставляют пользователям несколько путей для доступа к часто используемым функциям. Альтернативные пути также имеют смысл при наличии ошибок классификации, когда пользователь ищет функцию не в той категории. Несомненно, рекомендуется избегать таких неоднозначных классификаций, но на практике редко удаётся сформировать идеальную структуру. В этом случае можно дублировать функцию в нескольких категориях, избавив пользователя от ошибочных решений. С другой стороны, копии функции занимают дополнительные позиции в панелях меню и усложняют его структуру. Кратко обсудим, как описанная выше модель может быть расширена для решения этой дилеммы.

Рассмотрим множество функций N, которые необходимо разместить в меню. Каждой функции $w \in N$ поставлена в соответствие её популярность $\mu(w)$ – априорная вероятность того, что пользователь будет искать функцию w. Популярности определяются на основе статистики или экспертных мнений.

Определим нечёткую классификацию функций из N. Рассмотрим набор q категорий и для каждой функции $w \in N$ и каждой категории $l = 1, \ldots, q$ определим степень принадлежности $\theta_{wl} \in [0, 1]$. Предположим, что степени нормализованы, т.е. $\sum_{l=1}^{q} \theta_{wl} = 1$. Нечёткая классификация может строиться экспертом или вычисляться математическими методами, например, с помощью латентно-семантического анализа [12].

Чёткая структура формируется из нечёткой классификации путём деления функций. Если функция $w \in N$ принадлежит k категориям с ненулевой достоверностью, то она заменяется на k функций, каждая из которых принадлежит только одной категории. Популярность таких функций вычисляется умножением популярности исходной функции на степень её принадлежности соответствующей категории. Подобным образом преобразуются все доступные классификации.

Чёткие классификации далее могут быть использованы как описано в предыдущих разделах с незначительными изменениями в алгоритме. Если две копии разделённой функции оказываются в одной панели меню, то их необходимо объединить для минимизации времени поиска. Кроме того, когда оценивается время навигации в подменю, считается, что все копии одной функции попадают в одну панель меню, т.е. все копии должны быть объединены перед вычислением оценки.

Таким образом, нечёткие классификации позволяют моделировать альтернативные пути к одним и тем же функциям. Но всё равно остаётся открытым вопрос, в каких случаях стоит дублировать функции в нескольких категориях, а когда эффективнее избавиться от дубликатов, увеличив вероятность ошибки пользователя. Для решения этой проблемы можно предложить простой подход, который, однако, не гарантирует глобальной оптимальности решения. Фиксируем функцию и, используя предложенные выше алгоритмы, строим оптимальную иерархию для двух случаев: когда функция разделена по нескольким категориям, и когда функция принадлежит только одной категории в каждой классификации. Выбираем иерархию с меньшим временем поиска и переходим к следующей функции.

Заключение

Обзор публикаций в области построения иерархических меню демонстрирует, что существующие работы либо игнорируют семантические аспекты построения меню, либо не предлагают алгоритмов оптимизации структуры меню. Предложенный подход вводит понятие семантического качества в математическую модель оптимизации иерархических меню.

Модель оптимизации с учётом семантического качества требует разработки алгоритма оптимизации, учитывающего ряд факторов. Предложенный алгоритм использует ряд эвристических методов, позволяющих применять математическую модель на практике.

Предложенный алгоритм наряду с другими ручными и полуавтоматическими средствами построения меню был реализован в инструменте оптимизации иерархических меню. Инструмент был опробован на реальных меню различных типов и показал хорошие результаты работы.

Тем не менее, в задаче разработки меню, особенно с учётом семантических аспектов, имеется ещё немало открытых вопросов для дальнейшего исследования. Хотелось бы надеяться, что описанные в статье математическая модель и алгоритмы смогут составить основу для решения и этих проблем.

Литература

1. *Thimbleby H.* Press On: Principles of Interaction Programming. – MIT Press, 2007. – 528 p.
2. *Fu W.-T., Pirolli, P.* SNIF-ACT: a cognitive model of user navigation on the world wide web // Human-Computer Interaction. – 2007. – Vol. 22, N 4. – P. 355–412.
3. *Kitajima M., Blackmon M.H., Polson P.G.* A comprehension-based model of web navigation and its application to web usability analysis // Proc. of CHI 2000. – 2000. – P. 357–373.
4. *Miller C.S., Remington R.W.* Modeling Information Navigation: Implications for Information Architecture // Human-Computer Interaction. – 2004. – Vol. 19. – P. 225–271.
5. *Norman K.L.* The Psychology of Menu Selection: Designing Cognitive Control at the Human/Computer Interface. – Ablex Publishing Corporation, 1991.
6. *Snowberry K., Parkinson S.R., Sisson N.* Computer display menus // Ergonomics. – 1999. – Vol. 26. – P. 699–712.
7. *Lee E., MacGregor J.* Minimizing user search time in menu retrieval systems // Human Factors. – 1985. – Vol. 27, N 2. – P. 157–162.
8. *Landauer T.K., Nachbar D.W.* Selection from alphabetic and numeric menu trees using a touch screen: depth, breadth and width // Proc. of the SIGCHI conf. on Human Factors in Computing Systems. – 1985. – P. 73–78.
9. *Cockburn A., Gutwin C., Greenberg S.A.* Predictive Model of Menu Performance // Proc. of ACM CHI'07. – 2007. – P. 627–636.
10. *Card S.K., Moran T.P., Newell A.* The psychology of human-computer interaction. – Lawrence Erlbaum Associates, 1983.

11. *Soto R.* Learning and performing by exploration: Label quality measured by latent semantic analysis // Proc. of CHI'99 Human Factors in Computing Systems. – 1999. – P. 418–425.
12. *Landauer T.K., Dumais S.T.* A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge // Psychological Review. – 1997. – Vol. 104. – P. 211–240.
13. *Blackmon M.H., Kitajima M., Polson P.G.* Repairing usability problems identified by the cognitive walkthrough for the web // Proc. of the conf. on Human factors in computing systems. – 2003. – P. 497–504.
14. *Kuniavsky M.* Observing the User Experience: A Practitioner's Guide to User Research. – Morgan Kaufmann, 2003.
15. *Miller C.S., Fuchs S., Anantharaman N.S., Kulkarni P.* Evaluating Category Membership for Information Architecture. – DePaul CTI Technical Report 07-001, 2007.
16. *Fisher D.L., Yungkurth E.J., Moss S.M.* Optimal menu hierarchy design: syntax and semantics // Human Factors. – 1990. – Vol. 32, N 6. – P. 665–683.
17. *Губко М.В., Даниленко А.И.* Математическая модель оптимизации структуры иерархического меню // Проблемы управления. – 2010. – № 4. – С. 49–58.
18. *Goubko M.V., Danilenko A.I.* An automated routine for menu structure optimization // Proc. of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems. – 2010. – P. 67–76.
19. *McNaughton R.* Scheduling with Deadlines and Loss Functions // Management Science. – 1959. – Vol. 6, N 1. – P. 1–12.

DRAFT статьи, опубликованной в журнале «Проблемы управления», 2012, №2, С. 53-63.

Semantic-Aware Optimization of User Interface Menus

Goubko M.V., Danilenko A.I.

While the problem of hierarchical menu design is very common in user interface design, existing approaches lack either semantic aspects or optimization techniques. We suggest a semantic-aware mathematical model of hierarchical menu optimization and algorithms developed on the basis of this theory. These algorithms are implemented in the ready-to-use design tool. The approach is illustrated by the optimization of a banking voice menu.

Keywords: *user interface optimization, hierarchical menu, semantic quality, menu design automation*