Trois méthodes pour la construction de pages WEB personnalisées

Gérard KUBRYK
gerard@i2m.fr

Université de Nice-Sophia Antipolis, Laboratoire I3S, 2000, route des lucioles
06903 Sophia Antipolis cedex, France

Mots clefs:
Services Web et audio, analyse des besoins,pages personnalisées, spécifications de modèles et de méthodes, apprentissage

Keywords:
Web and audio services, requirements analysis, personalized and customized pages, models and methods specifications, machine learning

Palabras clave:
Servicios Web y audio, análisis de requisitos, ajuste de llamada, especificatión de modelos y méthodos, máquinas de aprendizaje

Résumé

Les techniques WEB ont pris une grande importance ces dernières années. Les analyses des chercheurs ont montré qu'il est nécessaire d'avoir de nouvelles méthodes pour permettre le développement des sites WEB et offrir aux utilisateurs des pages adaptés à leurs besoins.
Les services WEB et audio doivent offrir un service aussi bon que possible pour être attractifs. Pour ce faire, ils doivent être capables d'analyser les actions des utilisateurs sans porter atteinte à leur vie privée. Ce papier décrit deux classes de modèles, mathématiques et apprentissage, et trois méthodes possibles pour gérer et créer des pages personnalisées.

1 Introduction

Quand un utilisateur accède à un site WEB ou utilise un service à valeur ajoutée, il recherche un accès efficace pour obtenir les informations le plus vite possible. Une méthode pour atteindre cet objectif est de mémoriser ses actions antérieures et de là, lui offrir une page avec une organisations des items et une méthode d'accès qui soient fidèles à ses préférences, lui évitant ainsi toute perte de temps. Il faut aussi analyser les relations entre les items. Ils peuvent être proches et se renforcer l'un l'autre, ou peuvent être éloignés et s'affaiblir.

Pour une analyse marketing efficace, un prestataire de services doit connaître la position de son service dans sa courbe de vie. Les informations à utiliser sont les mêmes et nous allons voir que les outils peuvent également être les mêmes.

Les actions de l'utilisateur se définissent par l'âge de l'action, sa durée, son taux de répétition et sa période de répétition. L'usage et la gestion de ces informations ne doivent cependant pas porter atteinte à la vie privée des utilisateurs du service. Enfin, de nouveaux services doivent pouvoir être créés sans constituer une charge d'exploitation.

Le problème est de décrire un phénomène qui est de la forme 1/t mais a 2 équations :

- Une équation à l'origine
- Une équation après la première consultation

Par ailleurs la psychologie nous donne 2 règles de base :

- l'intérêt est fonction inverse du temps entre les deux derniers usages.
- l'intérêt pour un objet est une fonction de la répétition de l'usage

Ceci peut être décrit par un modèle additif conforme à la théorie de Bellman :

$$E_{(n)} = f(E_{(n-1)}, \frac{1}{t_n - t_{n-1}})$$

E_n (état à l'instant n) est fonction de E_{n-1} et de $\frac{1}{t_n - t_{n-1}}$.

Pour ce faire, nous allons comparer 3 modèles :

- analogie avec la gravité (métaphore de la « fontaine d'informations »)
- analogie avec les fourmis
- apprentissage par sanction-renforcement

Ces méthodes seront comparées selon 2 critères, leur aptitude à créer la bonne organisation des items et leur aptitude à gérer un nombre important de requêtes en temps réel. La dernière étape sera une expérimentation de la méthode choisie avec des utilisateurs réels, mais avant nous avons définit des utilisateurs « typiques » avec des stratégies prédéfinies que nous pourrons simulés. Ceci doit permettre de dégrossir les paramètrages des modèles et un gain de temps significatif sur l'expérimentation.

Les recherches dans ce domaine utilisent essentiellement des méthodes statiques. Elles consistent à définir un « utilisateur moyen » ou des catégories d'utilisateurs, c'est le cas par exemple des travaux de M. Perkowitz qui fait appel à des méthodes statistiques lourdes ne permettant pas de travailler en temps réel. Le propos de cette recherche est différent puisque l'objectif est de créer des pages individualisées pour chaque utilisateur.

Le système décrit est un cas particulier des systèmes d'apprentissage. L'activité de l'utilisateur laisse une trace utilisée pour construire une page qui est le résultat d'un d'apprentissage. Notre particularité est d'étudier chaque utilisateur individuellement et que l'objet de l'apprentissage est toujours une page (un ensemble d'items ordonnés)

L'architecture pour réaliser cette fonction est relativement simple. Elle a été décrite dans [8] G. Kubryk (2004)

2 Les modèles

2.1 L'analyse mathématique

Le problème posé est un problème d'organisation en fonction du temps. Ce temps est fonction de la façon dont l'utilisateur accède aux services proposés. Les informations vont être organisées en un vecteur qui permettra le classement des items dont la page sera tirée.

User ID **Item ID**	Ces 2 éléments servent de clés d'accès au vecteur
Date/heure	C'est la base de tous les calculs
f(t)	Cette valeur est la fonction du temps utilisée pour calculer le rang de l'item pour l'utilisateur
f'(t)	C'est la dérivée de la fonction ci-dessus (sa pente). Elle constitue une bonne méthode d'analyse de l'intérêt de l'item pour l'utilisateur - si la pente est positive et forte, l'intérêt est important - si la pente est positive ou négative mais faible, l'intérêt décroît - si la pente est négative et forte, l'intérêt est faible

Le système va créer un vecteur pour chaque utilisateur à chaque accès au service. Ce vecteur $\vec{X_i}$ contient les informations décrites ci-dessus et est utilisé pour calculer un nouveau vecteur résultant avec le ou les vecteurs précédents.

L'ensemble des vecteurs des items permet la création de la page.

$\vec{X_i} \xrightarrow{f(t)}$ Organisation des items

(f(t) est la fonction qui modifie $\vec{X_i}$ selon le temps entre 2 accès)

Le vecteur $\vec{X_i}$ peut prendre 2 formes selon que nous avons 1 vecteur par accès (n vecteurs par item) ou un seul (1 vecteur par item) qui est calculé à chaque accès.

Dans chaque cas, les vecteurs doivent comprendre :

- User ID
- URL ou item ID
- Horodatage de l'évènement
- Durée de l'évènement

La forme « 1 vecteur par item » nécessite en plus :

- f(t) calculé à l'instant de l'accès pour le couple utilisateur-item
- f'(t) calculé à l'instant de l'accès pour le couple utilisateur-item

Les vecteurs X_i sont :

N vecteurs par item $\begin{bmatrix} \text{User Id} \\ \text{Item ID} \\ \text{date / heure} \\ \text{durée} \end{bmatrix}$

1 vecteur par item $\begin{bmatrix} \text{User Id} \\ \text{Item Id} \\ \text{date / heure} \\ \text{durée} \\ \text{f(t)} \\ \text{f' (t)} \end{bmatrix}$

Le choix entre ces 2 formes a un effet important sur le temps de réponse du système. Dans le cas « n par item » il faut recalculer toute la chaîne des évènements pour obtenir le f(t) et le f'(t) final. Ce temps croit obligatoirement au fil du temps en fonction du nombre de consultations du service. La solution « 1 par item » est en ce sens plus efficace puisque le temps de calcul est le même à chaque consultation quoi qu'il arrive. Enfin, il garantit pleinement la vie privée de l'utilisateur, car le calcul n'est pas réversible.

La même méthode peut être appliquée à un item ou une URL en cumulant les accès de l'ensemble des utilisateurs. En ce cas le vecteur devient : $\begin{bmatrix} \text{Item Id} \\ \text{date / heure} \\ \text{durée} \\ f(t) \\ f'(t) \end{bmatrix}$

Il n'y a pas de « user ID » car il n'a pas de sens en ce cas. Ceci permet d'avoir une analyse en temps réel d'un item et de déterminer sa position dans sa courbe de vie.

2.1.1 L'analogie avec la gravité

Une fonction f(t) peut être la gravité. En appliquant cette analogie à différents items, il est possible de leur calculer une altitude différente, fonction de l'activité de l'utilisateur, ce qui donnera la place dans la page. L'analogie est la « fontaine » ou chaque goutte à un temps de vol durant lequel l'item est visible dans la page.

Cette « fontaine d'informations » utilise des lois physiques élémentaires :

- L'information est définie comme un « quantum » de masse M.
- Cette masse reçoit une quantité d'énergie qui lui donne une vitesse verticale selon la loi : $E = ½\ m\ v^2$
- La vitesse évolue selon : $V(t) = V_0 - g\ t$
- L'altitude maximum P_m est : $P_m = ½\ \frac{V_0^2}{g}$

En ce cas, l'équation de la vitesse devient : $V_r(t) = V_0 + V_1 - g\ t$
et l'altitude : $P_r(t) = V_0 t - ½\ g\ t^2 + V_1\ (t - t_1)$

Les équationsgénérales pour n apports d'énergies sont
pour la vitesse : $V(t) = \sum_{i=0}^{n} V_i - g\ t$ et pour l'altitude : $P(t) = \sum_{i=0}^{n} V_i\ (t - t_i) - ½\ g\ t^2$

2.1.2 L'analogie avec les fourmis

Ce modèle utilise une analogie avec la façon dont les fourmis agissent pour nourrir leur nid et résoudre les problèmes.

La fourmi est un insecte simple avec une mémoire limitée et seulement capable d'actions simples. Cependant, une colomie de fourmis est capable de comportements collectifs complexes apportant des solutions « intelligentes » aux problèmes.

Une fourmi isolée n'a pas de connaissance globale des tâches à accomplir. Les actions des fourmis sont des décisions locales et sont de ce fait imprédictibles. Le comportement intelligent émerge de l'auto-organisation et de la communication entre les fourmis. Ceci est appelé *intelligence émergente* ou *comportement émergent*.

Pour certains scientifiques, la conscience de l'homme est une forme de comportement emergent provenant de l'interaction des neurones.

Une fourmi qui se déplace laisse une petite quantité de phéromone. Les phéromones sont des produits chimiques servant à la communication avec les autres fourmis. Cette pheromone crée un chemin permettant à la fourmi de retrouver son nid. C'est aussi le chemin par lequel une fourmi retrouve une source de nourriture une fois recrutée.

Plus les fourmis utilisent un chemin plus la quantité de phéromone croit. Les fourmis préferent le chemin qui a le plus de pheromone. Pour une importante source de nourriture, de nombreuses fourmis sont recrutées, il se crée une sorte d'autoroute, que de plus en plus de fourmis utilisent.

Il a une petite quantité de fourmis avec un “programme différent”. Elles explorent d’autres chemins et peuvent devenir recruteur pour une nouvelle source de nourriture.

La phéromone s’évapore, un chemin non utilisé devient de moins en moins attractif.

Dans cette analogie, les fourmis sont les consultations, le nid est l’utilisateur et la nourriture est l’ensemble des items. Le niveau de pheromone est le rang de l’item.

Nous disposons de tout les elements pour la création de pages WEB.

- l’horloge est donnée par l’évaporation de la phéromone
- l’accroissement du niveau de pheromone pour chaque consultation
- les fourmis atypiques donnent l’accès aux items nouveaux et non utilisés

Une équipe de l’Université Paul Sabatier à Toulouse [4] (Dussutour and Al) a montré que l’évaporation à une loi très simple, 1/40.

Ils ont surtout montré que sur un chemin surchargé, les fourmis savent créer un autre chemin et se répartir entre les deux chemins. Ceci ouvre, pour notre problème, le concept de synergie qui permet de partager les variations liées aux actions antérieures de l’utilisateur avec les items proches.

Cette proximité est définie par deux sources:

- les déclarations de l’utilisateur à son inscription au fournisseur d’accès
- le cheminement de l’utilisateur dans les pages qui définit des proximités

La proximité peut se définir par une matrice qui par exemple pour l’item A serait:

Item	Consultation	Pourcent
A	35	42,7
B	27	32,9
C	12	14,6
D	8	9,6
Total	82	99,8

Le partage de l’apport peut être pratiqué à partir d’un certain niveau de l’item. La règle de partage proposée est une proportion du nombre de consultations. Cette proportion doit être telle que l’item « principal » est favorisé et croit plus vite que le ou les items proches.

La formule est : **Total / Nb consultations * 100**. Ce qui nous donne le pourcentage de répartition de la croissance pour l’item A et pour les items proches.

2.2 L’apprentissage

L’apprentissage est un autre moyen de définir la position d’un item dans la page. Différents moyens d’introduire l’apprentissage dans les systèmes informatiques ont été décrits

La psychologie définit l’interaction avec l’homme ou l’animal pour créer une nouvelle aptitude comme « l’apprentissage ». L’apprentissage a différentes formes. L’une d’elle est l’apprentissage par renforcement, une action récompensée est renforcée (est plus probable) une action non récompensée est affaiblie (est moins probable).

Ce renforcement est modifié par différents facteurs, l’habitude (quand le résultat est proche de ce qui est attendu) qui affaiblit le renforcement, la surprise (quand le résultat est inattendu) qui augmente le renforcement. Enfin l’apprentissage est sujet à l’oubli

La base du système proposé est le rang de l’item sélectionné par l’utilisateur à l’instant « t ». Ce rang est dans l’ordre naturel, du rang 1 au rang « n », n est limité par l’écran utilisé comme référence par le concepteur de la page et la lisibilité. Ceci fait que la différence de rang est négative quand l’utilisateur va du rang « n » vers le rang 1.

Ce dont nous avons besoin dans notre système est : la prédiction précédente, le choix précédent de l'utilisateur (ceci nous donne la différence entre les deux) et la différence de temps entre les deux derniers choix. La formule générale est alors:

$$P_n = g(P_{n-1}, Choix_{n-1}, \Delta, \Delta_t)$$

Dans cette formule:

P est la prédiction du système

Choix est le rang de l'item choisit par l'utilisateur

Δ est la différence entre la prédiction précédente et le choix de l'utilisateur

$P_{n-1} - Choix_{n-1}$

Δ_t est la différence de temps entre les deux derniers accès

g est la fonction liant tous ces facteurs

Le premier paramètre que nous allons définir est l'oubli (Kf). Ce paramètre définit la façon dont le système oubli ce qu'il a prédit. Ceci signifie que si Kf décroît (augmentation de l'oubli) le poids du choix de l'utilisateur s'accroît. Notre formule devient:

$$P_n = K_f P_{n-1} + (1 - K_f) Choix_{n-1} + f(\Delta, \Delta_t)$$

K_f agit de la façon suivante:

Si $K_f = 0$ (oubli maximum) notre formule devient: $P_n = Choix_{n-1} + f(\Delta, \Delta_t)$

Le système a complètement oublié ses prédictions et seul le choix de l'utilisateur modifie la prédiction. Ceci est tempéré par le facteur $f(\Delta, \Delta_t)$.

Si $K_f = 1$ (aucun oubli) notre formule devient: $P_n = P_{n-1} + f(\Delta, \Delta_t)$

Le système n'oublie pas et les choix de l'utilisateur n'ont aucune influence. Ceci est tempéré par le facteur $f(\Delta, \Delta_t)$.

Pour que la formule soit correcte il est nécessaire que $f(\Delta, \Delta_t)$ puisse être égal à 0 quand Kf = 0 et quand Kf = 1. Ceci implique que « f » est une courbe $p(K_f)$ à définir qui coupe l'axe des « X » pour ces deux valeurs. Pour ces deux cas la formule devient:

Si $K_f = 0$ alors $P_n = Choix_{n-1}$

Le système oublie ses prédictions et seul l'utilisateur a une influence.

Si $K_f = 1$ $P_n = P_{n-1}$

Le système n'oublie rien et n'évolue pas selon les choix de l'utilisateur.

La formule générale devient:

$$P_n = K_f P_{n-1} + (1 - K_f) Choix_{n-1} + p(K_f)(\Delta, \Delta_t)$$

Nous devons introduire 3 autres paramètres, le premier est la surprise, le second l'habitude et le dernier le renforcement.

Ces facteurs sont décrit par $p(K_f)(\Delta, \Delta_t)$

La surprise accroît l'apprentissage.

L'habitude décroît l'apprentissage.

Ainsi la surprise et l'habitude sont les deux faces du même paramètre que nous appelons K_{sa}. Nous disons qu'il y a habitude si la différence entre le rang prédit et le rang réel est inférieure à « X » et surprise quand la différence est supérieure à « X ».

$\Delta > X$ surprise, $\Delta < X$ habitude, $\Delta = X$ indifférence

Par convention le facteur K_{sa} varie entre 0 et 2 et suit la règle suivante

$0 > K_{sa} > 1$ habitude, $K_{sa} = 1$ indifférence, $1 > K_{sa} > 2$ surprise

Le renforcement est fonction de $1/\Delta_t$. Une période courte entre deux accès signifie un intérêt pour cet item, ce paramètre est K_r .

Le facteur $p(K_f)(\Delta, \Delta_t)$ que nous appelons sanction a donc la forme suivante:

$p(K_f)(K_{sa} \Delta + K_r \Delta / \Delta_t)$ et après factorisation : $p(K_f) \Delta (K_{sa} + K_r / \Delta_t)$

La formule complète est:

$$\mathbf{P_n = K_f P_{n-1} + (1 - K_f) Choix_{n-1} + p(K_f) \Delta (K_{sa} + K_r / \Delta_t)}$$

3 Méthode d'évaluation

Une étape importante est l'évaluation des différents algorithmes et leur paramétrage. Pour pouvoir le faire dans des conditions comparables nous avons défini des utilisateurs standardisés ayant diverses stratégies

Ces stratégies reposent sur 3 paramètres

- l'interval de temps entre 2 consultations
- la diversité dans le choix des items
- la fidèlité dans les choix de l'utilisateur

Ces 3 paramètres nous donnent 8 types d'utilisateurs qui se définissent comme suit :

- Internaute actif, stable (Fréquence élevée, diversité élevée, fidélité élevée)
- Internaute actif instable (Fréquence élevée, diversité élevée, fidélité faible) (*pas d'objectifs*)
- Internaute actif ciblé stable (Fréquence élevée, diversité faible, fidélité élevée) (*objectifs précis*)
- Internaute actif ciblé instable (Fréquence élevée, diversité faible, fidélité faible) (*ce cas ne semble pas réaliste*)
- Internaute faiblement actif stable (Fréquence faible, diversité élevée, fidélité élevée) (*consultant occasionnel fidèle*)
- Internaute faiblement actif instable (Fréquence faible, diversité élevée, fidélité faible) (*consultant touche à tout*)
- Internaute faiblement actif ciblé stable (Fréquence faible, diversité faible, fidélité élevée) (*objectifs précis peu actif*)
- Internaute faiblement actif non ciblé instable (Fréquence faible, diversité faible, fidélité faible)

Les valeurs de chacun des paramètres utilisés dans les simulations sont les suivantes aléatoirement :

Fréquence élevée $T_{n-1} - T_n \in [1,2,3]$
Fréquence faible $T_{n-1} - T_n \in [10,20,30]$

Si les items $i_j \in [i_1 - i_n]$
Pour la diversité élevée « j » varie de 1 à 20
Pour la diversité faible « j » varie de 1 à 10

Fidèlité élevée le rang varie de ± 2
Fidèlité faible le rang varie de ± 7

4 Conclusions

Dans ce papier nous avons présenté 2 classes de modèles (mathématique et apprentissage) et trois méthodes pour construire des pages adaptées pour le Web et les services audio.

La prochaine étape est avec la fin de définition des modèles, leur comparaison en terme d'efficacité (temps de réponse, charge de calcul) et de cohérence des résultats.

Ensuite viendra la vérification de la perception psychologique du modèle et sa cohérence avec les attentes des utilisateurs en même temps que la définition des paramètres à appliquer au modèle.

Bibliographie

[1] R. Baron, M.B. Gordon, H. Paugam-Moisy, et al. “Comparative study of three connectionist models on a classification problem”, Ecole Normale Supérieur de Lyon Laboratoire de l’informatique du Parralélisme, Research report, 1996

[2] P Collard , « L’apprentissage discriminant dans MAGE », Nice University / HDR, 1992.

[3] P Collard, C.Escazut, A. Gaspar, “An evolutionary approach for time dependant organization”, International Conference on tools for Artificial Intelligence 1996

[4] A.Dussutour, V. Fourcassie, D. Hebling, JL Deneubourf, 2004. “Optimal traffic organization in ants under crowded conditions”. Nature, Vol 428, 4 march 2004

[5] Alessio Gaspar, Philippe Collard “Immune Approaches to Experience Acquisition in Time Dependent Optimization” Artificial

Immune Systems, Las Vegas, Pages 49-50, 2000

[6] Alessio Gaspar, Philippe Collard, “Two Models of Immunization for Time Dependent Optimization”, 2000

[7] Alessio Gaspar, Philippe Collard “Time Dependent Optimization with a Folding Genetic Algorithm” Int. Conf. on Tools for Artificial Intelligence, IEEE Computer Society Press",207-214 1997

[8] Gérard Kubryk “Description de modèles pour des menus WEB adaptatifs”. CNRIUT 201-208 2004

[9] M. Perkowitz and O. Etzioni. “Adaptative Web Sites : Automatically synthetizing Web pages “. Proceeding of the fifteen National Conference on Artificial Intelligence 1998

[10] M. Perkowitz and O. Etzioni. “Adaptative Sites : Automatically learning from User access patterns” University of Washington, Department of Computer Science, Internal report

[11] Michèle Sebag, Marc Schoenauer, Caroline Ravisé “Inductive Learning of Mutation Step-size in Evolutionary Parameter Optimization”, Evolutionary Programming VI, 247-261, 1997

[12] Marlène Villanova-Oliver, Jérôme Gensel, Hervé Martin, « Progressive Access : A step toward adaptability in Web-based informations sytems”, OOIS 2002, LNCS 2425, 422-433, 2002