Adaptive User Interfaces for Personalized Services

Pat Langley

Center for Design Research
Stanford University, Stanford, California

and

Institute for the Study of Learning and Expertise
Palo Alto, California

http://www.isle.org/~langley

Background

The Need for Intelligent Assistance

As information and choices become more available, users need help in finding, and selecting among, the many alternatives.

This has led to the development of *recommendation systems,* which attempt to locate and recommend relevant items.

The Need for Personalized Assistance

At the same time, society is becoming ever more diversified.

Differences in private and professional preferences are growing.

Internet users are becoming increasingly selective about what they want to see and purchase.

We need *personalized* systems that can give users the information or product they want.

But personalized response requires some *model* or *profile* of the user.

Approaches to User Modeling

There are four distinct approaches to creating and utilizing user profiles for personalized services:

- manual creation by individual users (e.g., MyYahoo);
- manual creation of stereotypes and assignment of users based on demographic or behavioral data;
- offline learning of stereotypes from demographic/behavioral data and assigning users to them;
- online learning of individual user models from traces of their interactions.

We will refer to systems of the last sort as *adaptive user interfaces*.

The Problem of Learning Individual Models

We can state the problem confronting adaptive user interfaces as:

- *Given:* a set of tasks that require some user decision
- *Given:* descriptions for each of these tasks
- *Given:* traces of the user's decision on each task
- *Find:* mappings from task features to user decisions

There exist two broad approaches to describing the user's task:

- *collaborative* methods refer to other users' responses to the task
- *content-based* methods refer to measurable features of the task

Our work focuses on content-based approaches to user modeling.

Examples of Adaptive User Interfaces

Adaptive interfaces have been developed for many different tasks:

- Command and form completion
- Email filtering and filing
- News selection and layout
- Browsing the World Wide Web
- Selecting movies and TV shows
- On-line shopping
- In-car navigation
- Interactive scheduling
- Dialogue systems

These efforts cover a wide spectrum but also raise common issues.

Definition of an Adaptive User Interface

a software
artifact

that reduces
user effort

by acquiring
a user model

based on past
user interaction

Definition of a Machine Learning System

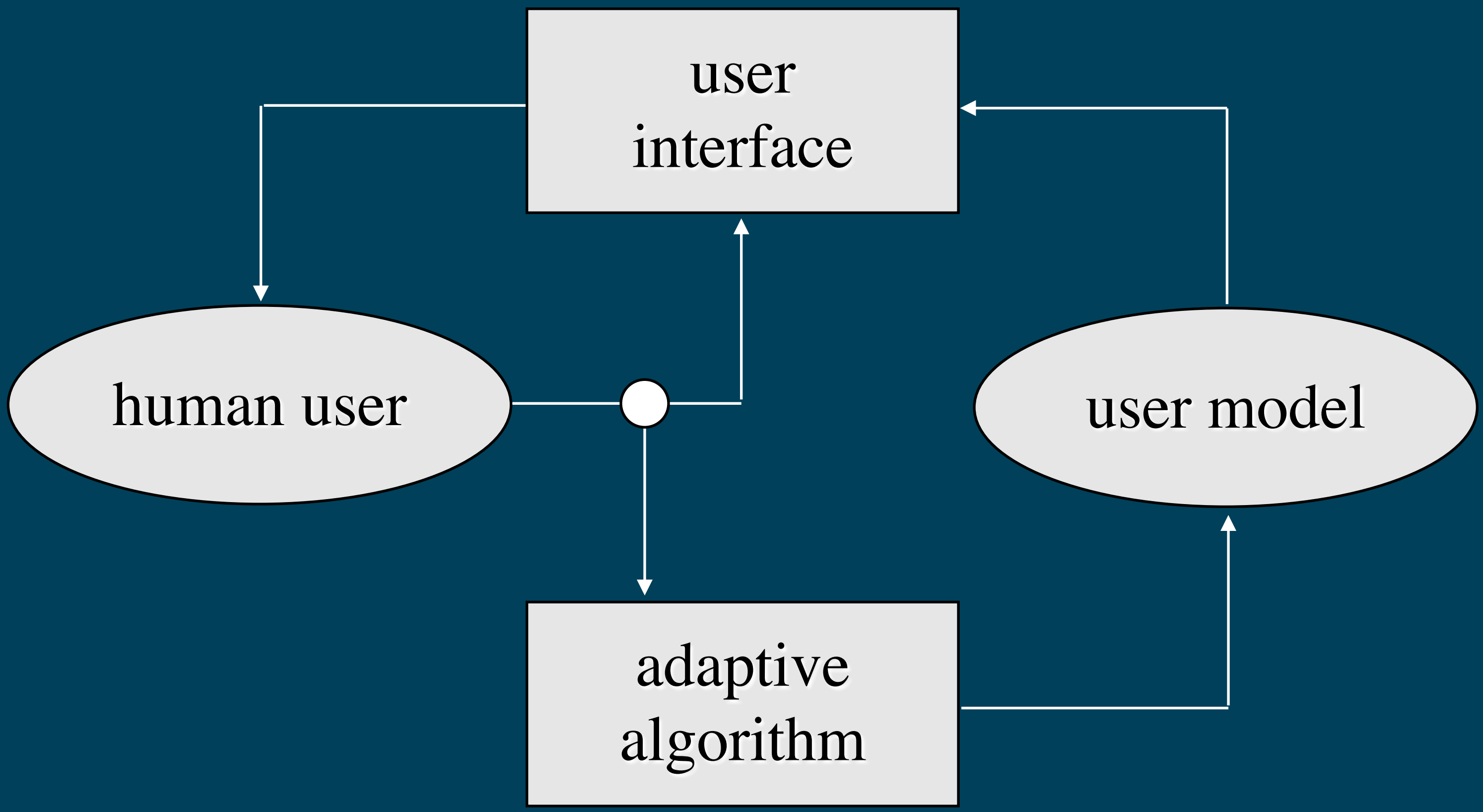
a software artifact

that improves task performance

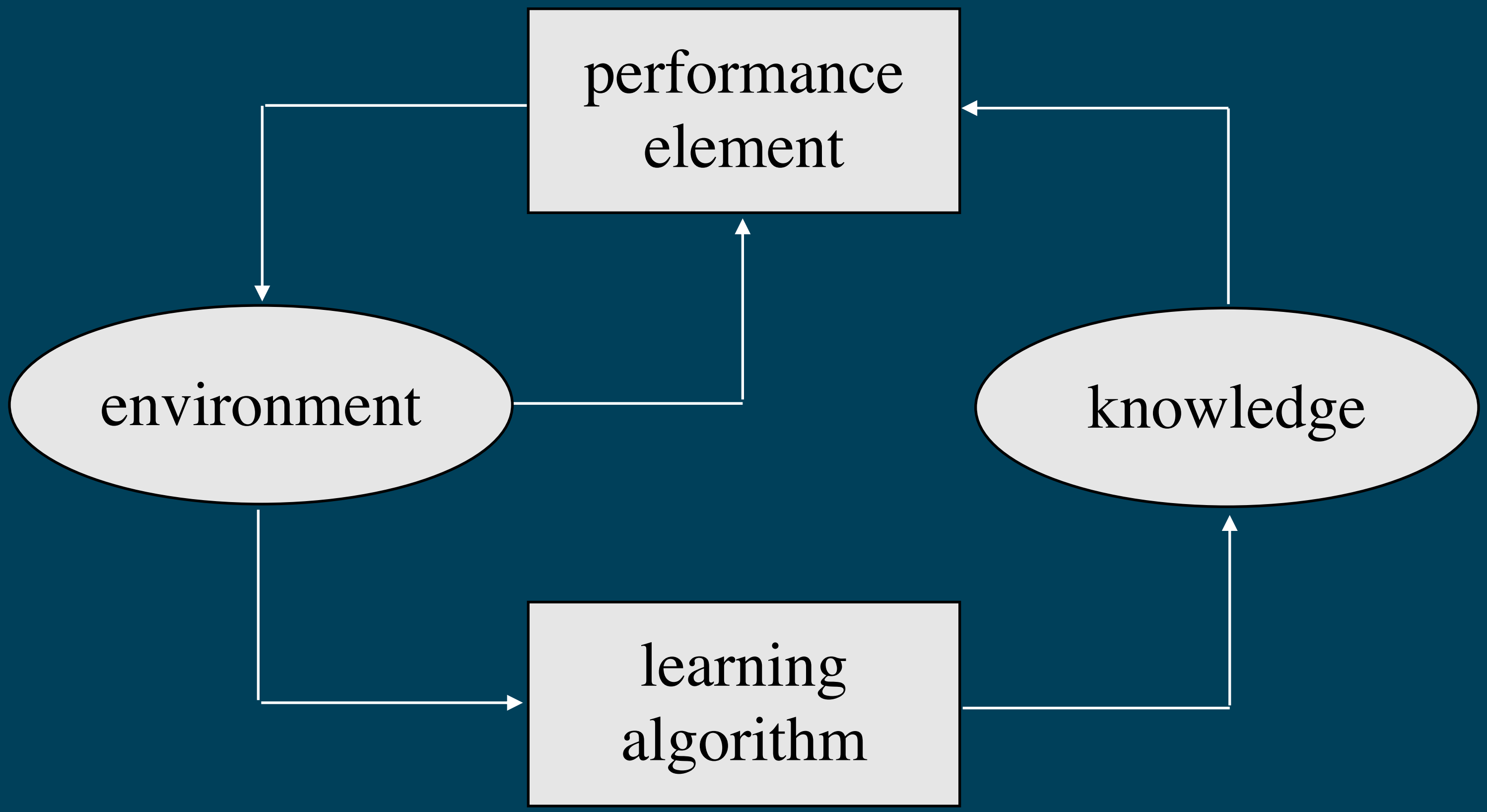
by acquiring knowledge

based on partial task experience

Elements of an Adaptive Interface

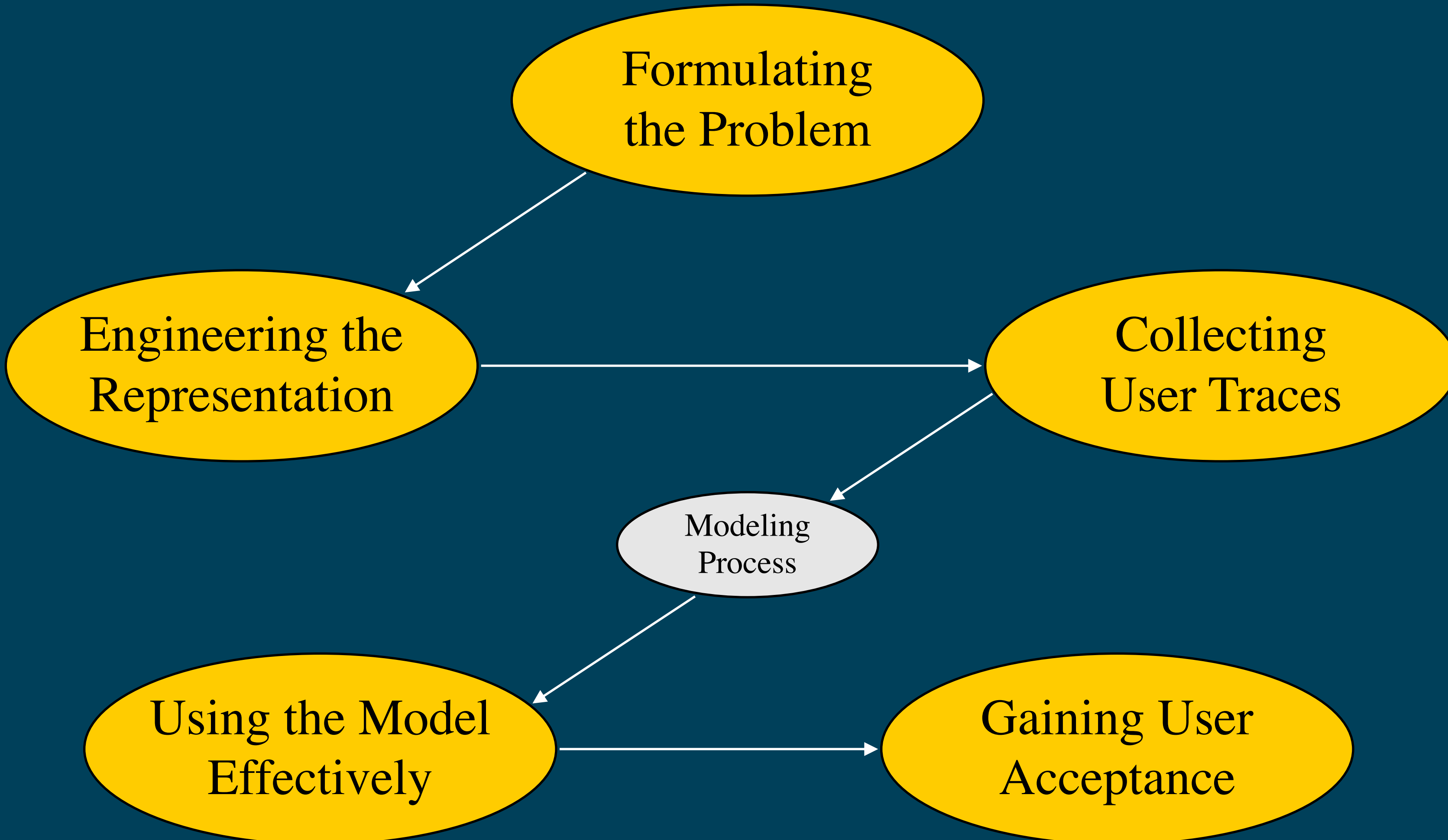


Elements of Machine Learning



Designing Adaptive User Interfaces

Steps in Developing an Adaptive Interface



Five Paradigms for Machine Learning

Rule Induction

Decision-Tree Induction

Case-Based Learning

Neural Networks

Probabilistic Learning

Problem Formulation

The first hurdle of an adaptive interface developer can be stated:

- *Given:* Some task that an intelligent assistant could aid;
- *Find:* Some formulation that lets the assistant improve its performance by learning a user model from experience.

This decision includes making clear design choices about:

- the aspect of user behavior to be predicted;
- the level of description (what constitutes a training case).

Since most robust learning methods focus on *supervised learning*, most adaptive interfaces formulate the task in these terms.

Representation Engineering

Another stage in developing an adaptive interface can be stated:

- *Given:* A formulation of some task in machine learning terms;
- *Find:* Some representation for behavior and user models that makes the learning task tractable.

This decision includes making clear design choices about:

- the information to be used when predicting behavior;
- the internal encoding of that information in the system.

Since most robust learning methods assume an *attribute-value* formalism, most adaptive interfaces take this approach.

Collecting User Traces

A third step in designing an adaptive interface can be posed as:

- *Given:* A problem formulation in terms of machine learning and a representation of user behavior;
- *Find:* An effective way to collect traces of this behavior.

This decision includes making clear design choices about:

- how to transform these traces into training instances;
- what action the user must take to generate the traces.

Since people seldom like extra burdens, an ideal adaptive interface requires *no extra user effort* to collect such traces.

Using the Learned Model

Another essential step in the development process can be stated:

- *Given:* An approach to learning a user model for some task;
- *Find:* Some way to invoke the model that helps the user perform the task more effectively.

This decision includes making clear design choices about:

- when and how to present the model's predictions to user;
- how to handle cases in which these predictions are wrong.

The ideal adaptive interface lets the user take advantage of good predictions and ignore bad ones.

Gaining User Acceptance

A final important facet of the development process can be stated:

- *Given:* A complete adaptive user interface for some task;
- *Find:* Ways to get people to try the system and to become long-term users.

Attracting first-time users involves marketing much more than technology, but, without it, a good system may be ignored.

However, a system that is well-designed and easy to use is more likely to retain users over long periods.

Examples of Adaptive User Interfaces

The Task of Route Selection

A decision-making task that confronts drivers can be stated as:

- *Given:* The driver's current location *C;*
- *Given:* The destination *D* that the driver desires;
- *Given:* Knowledge about available roads (e.g., a digital map);
- *Find:* One or more desirable routes from *C* to *D*.

Computational route advisors already exist in both rental cars and on the World Wide Web.

However, they do not give *personalized* navigation advice to individual drivers.

An Approach to Route Selection

Here is a one approach to learning route preferences, though not the first we considered:

- *Formulation:* Learn a “subjective” function to evaluate entire routes
- *Representation:* Global route features computable from digital maps
- *Data collection:* Preference of one complete route over another
- *Induction:* A method for learning weights from preference data
- *Using model:* Apply subjective function to find “optimal” route

This method learns a user model with respect to the *entire* route.

In this way, it avoids two important problems: *data fragmentation* and *credit assignment*.

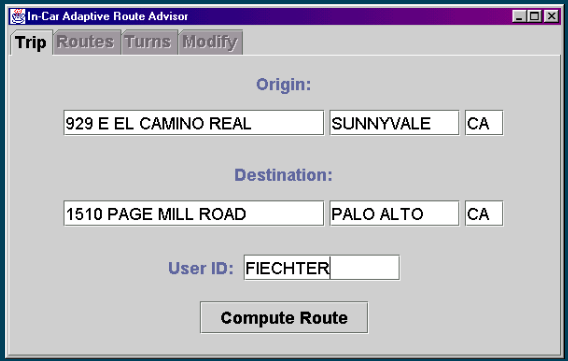
The Adaptive Route Advisor

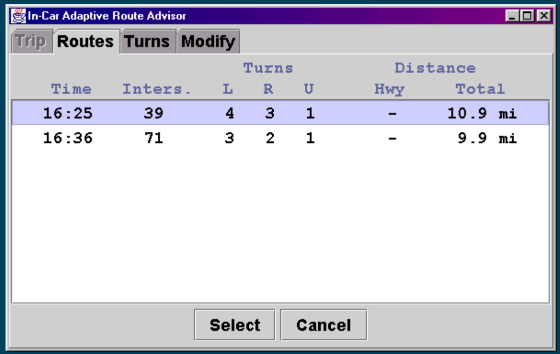
We incorporated these design choices into the *Adaptive Route Advisor* (Fiechter, Rogers, & Langley, 1999), which:

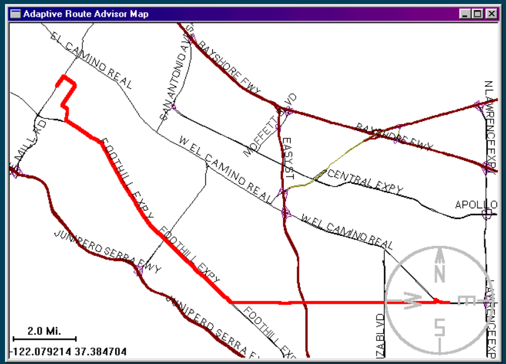
- models driver preferences in terms of 14 global route features
- gives the driver two *alternative* routes he might take
- lets the driver *refine* these choices along route dimensions
- uses driver choices to refine its model of his preferences
- invokes the driver model to recommend future routes

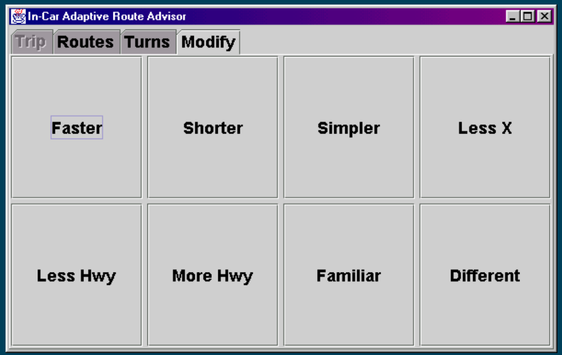
Note that providing drivers with choices lets the system collect data on route preferences in an unobtrusive manner.

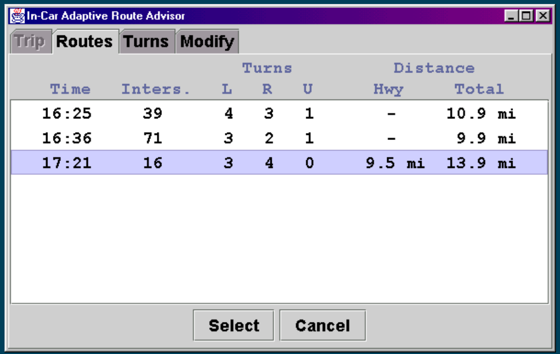
The Adaptive Route Advisor

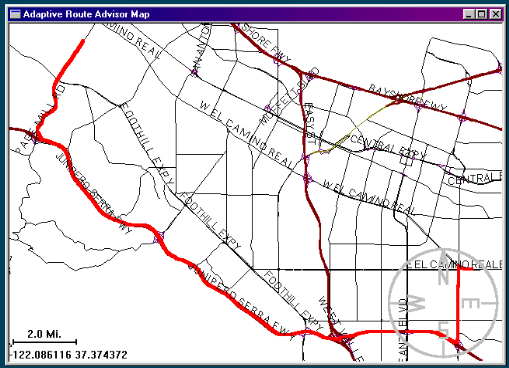






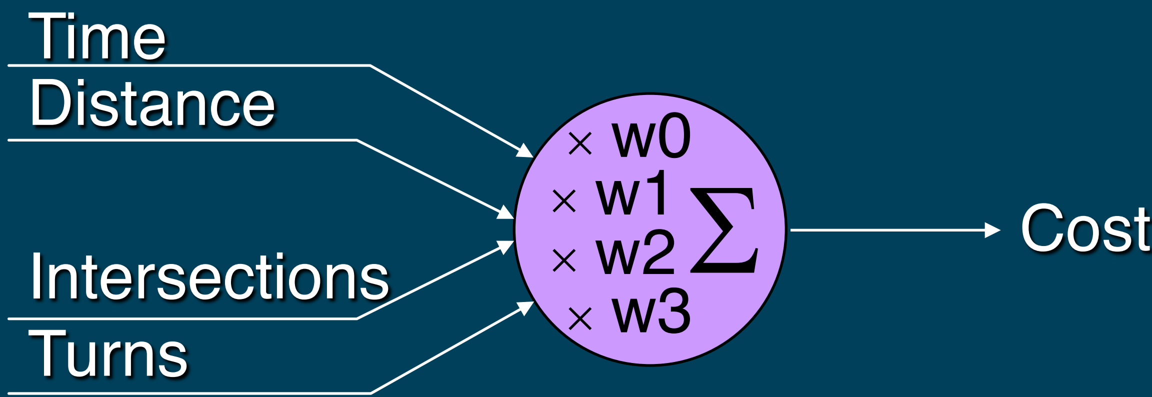






Driver Model and Training Cases

The Adaptive Route Advisor represents the driver model as a weighted linear combination of route features.

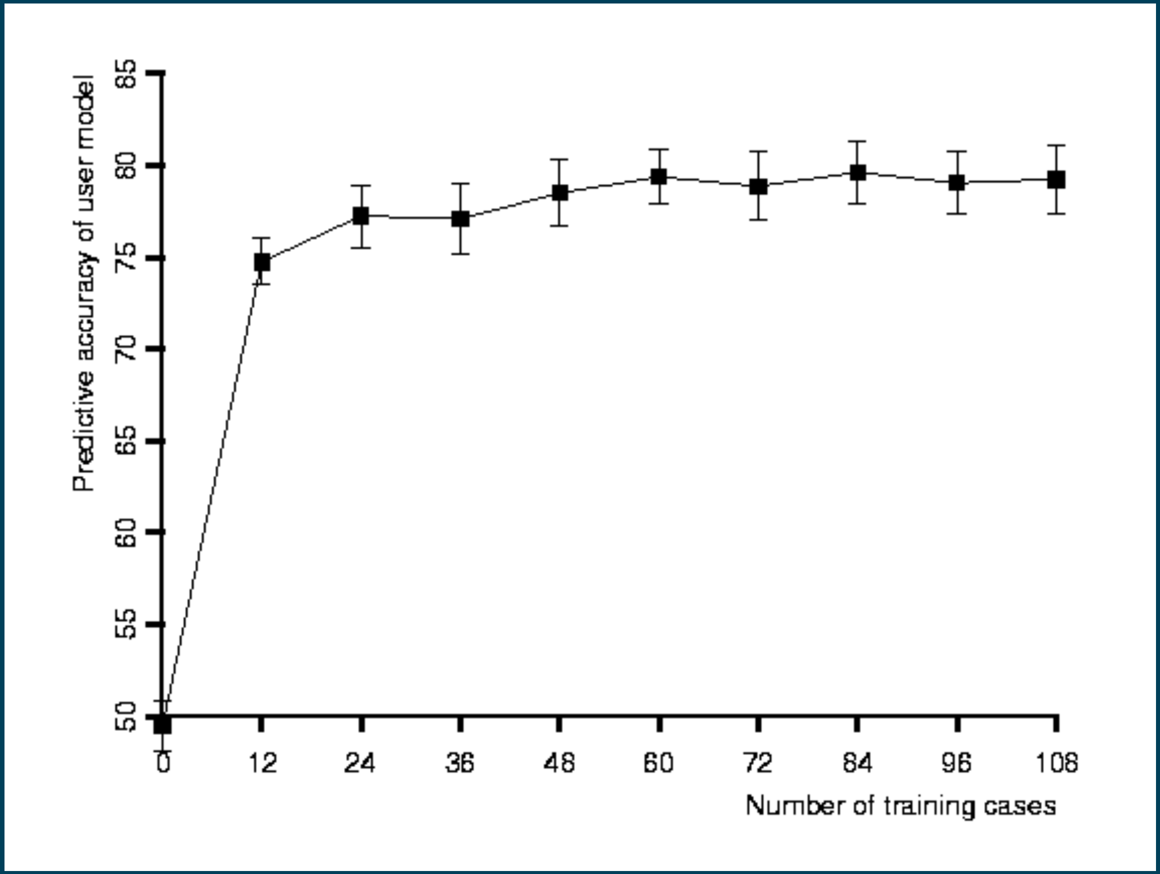


Training data: [x0, x1, x2, x3] is better than [y0, y1, y2, y3].

The system uses each training pair as constraints on the weights found during the learning process.

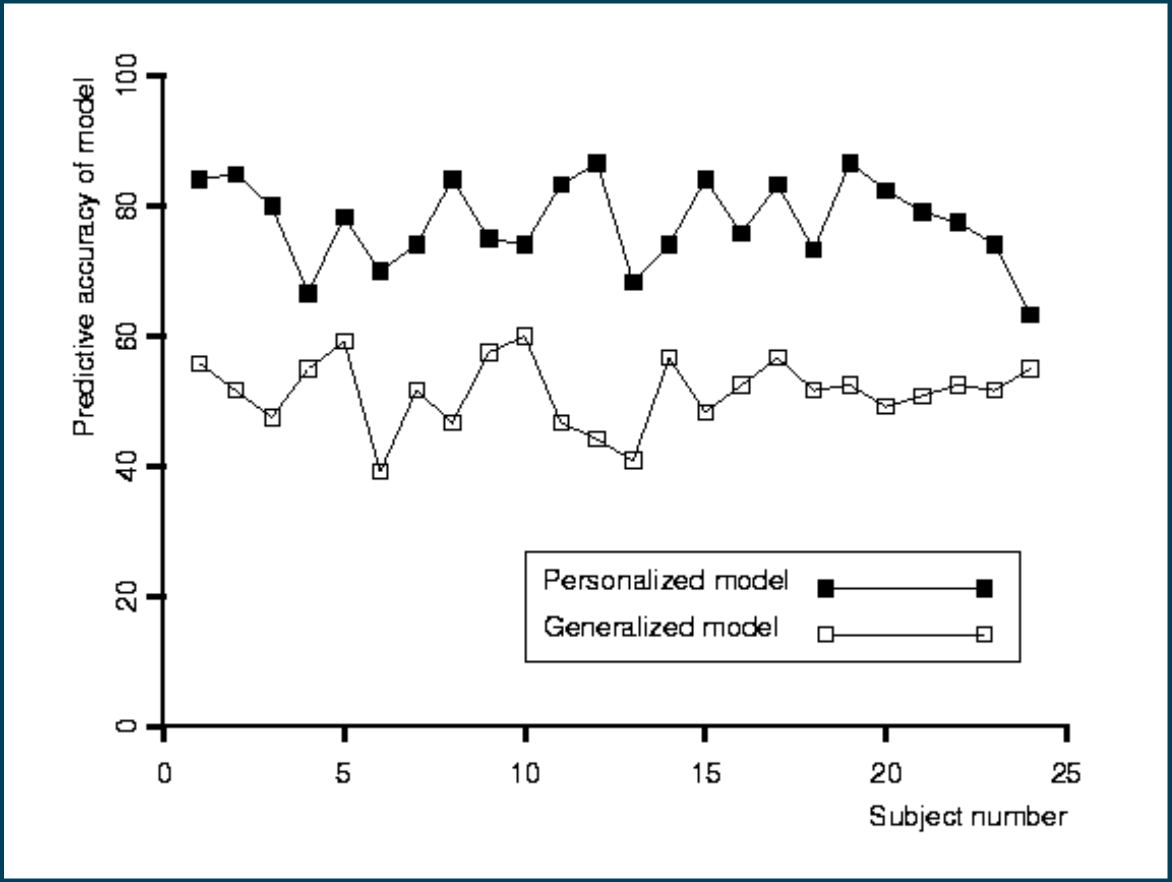
Experimental Results on Route Advice

Experiments with 24 subjects show the Route Advisor improves its predictive ability rapidly with experience.



Experimental Results on Route Advice

Analyses also show that personalized user models produce better results than generalized models, even when given more data.



The Task of Destination Selection

Aother type of decision that confronts drivers can be stated as:

- *Given:* The driver’s current location *C;*
- *Given:* Some goal the driver wants to achieve (e.g., eat lunch);
- *Given:* Knowledge about available nearby destinations;
- *Find:* One or more destinations *D* that achieve the goal.

Many recommendation systems are available on the World Wide Web, including ones that suggest restaurants.

However, they are ill suited for use in the driving environment, where visual attention is a limited resource.

Conversational Destination Advice

One approach to destination recommendation can be cast as:

- *Formulation*: Learn to predict features user cares about in items
- *Representation*: Conditions/weights on attributes and values
- *Data collection*: Converse with user to help him make decisions, noting whether he accepts or rejects questions and items
- *Induction*: Any supervised induction method
- *Using model*: Guide the dialogue by selecting informative questions and suggesting likely values

This design relies on the idea of a *conversational* user interface.

Spoken-language versions of this approach appear well suited to the driving environment.

The Adaptive Place Advisor

We implemented this approach in the *Adaptive Place Advisor* (Langley, Thompson, Elio, & Haddadi, 1999), which:

- engages in spoken conversations to help a user refine goals
- incorporates a dialogue model to constrain this process
- collects and stores traces of interaction with the user
- personalizes both its questions and recommended items

Our work focused on recommending restaurants to users who want advice about where to eat.

We believe this approach to recommendation will work well for drivers, but it also has broader applications.

A Sample Destination Dialogue

Inquirer: Where do you think I should eat tonight?

Advisor: Well, what type of cuisine would you like?

Inquirer: What types are there?

Advisor: Some examples are Chinese, Indian, and Mediterranean.

Inquirer: Oh, maybe Indian.

Advisor: What quality rating would you like?

Inquirer: I don't care, as long as it's cheap.

Advisor: How do you want to pay?

Inquirer: No, I think I'd like Chinese instead.

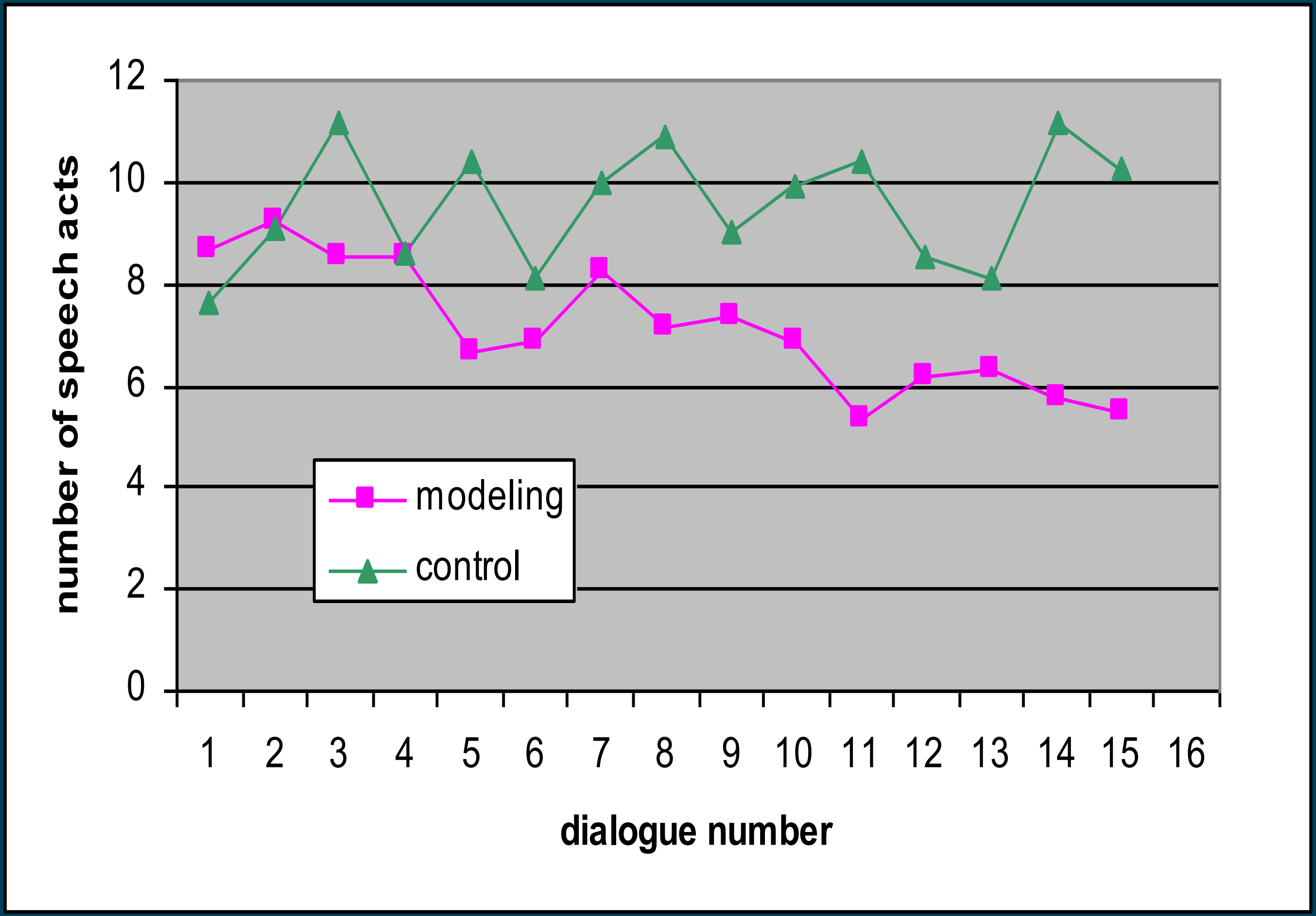
Advisor: Okay, we'll switch to Chinese food. What city do you prefer?

Inquirer: How about Berkeley?

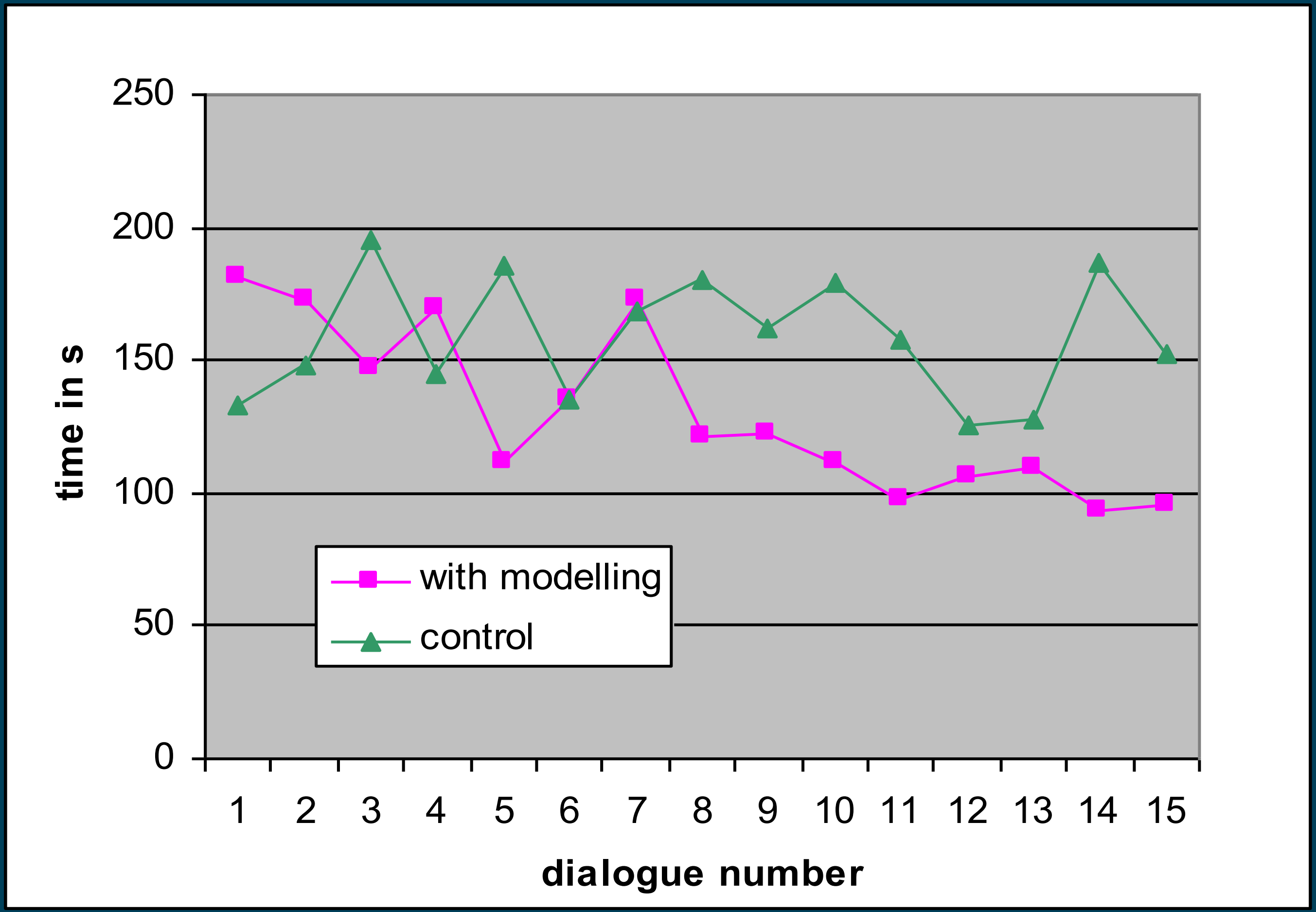
Advisor: I know three cheap Chinese restaurants in Berkeley. One is the Long Life Vegi House on 2129 University Avenue. Does that sound alright?

Inquirer: Sure, that sounds fine.

Speech Acts Per Conversation



Time Per Conversation



The Task of Resource Scheduling

Another complex problem that people encounter can be stated:

- *Given:* A set of tasks or jobs that must be done soon;
- *Given:* A limited set of resources available for these jobs;
- *Given:* Knowledge about constraints on jobs and resources;
- *Find:* One or more schedules that complete the jobs.

Many automated scheduling systems are used in industry, and some interactive schedulers exist.

But these systems do not generate *personalized* schedules that reflect the preferences of individual users.

An Approach to Personalized Scheduling

Here is one approach to preferences about learning schedules:

- *Formulation:* Learn a utility function to evaluate entire schedules
- *Representation:* Global features computable from the schedule
- *Data collection:* Preference of one candidate schedule over others
- *Induction:* A method for learning weights from preference data
- *Using model:* Apply 'subjective' function to find a good schedule

This method is similar to that in the Adaptive Route Advisor.

But it assumes search through a space of complete schedules (a *repair space*), which requires some initial schedule.

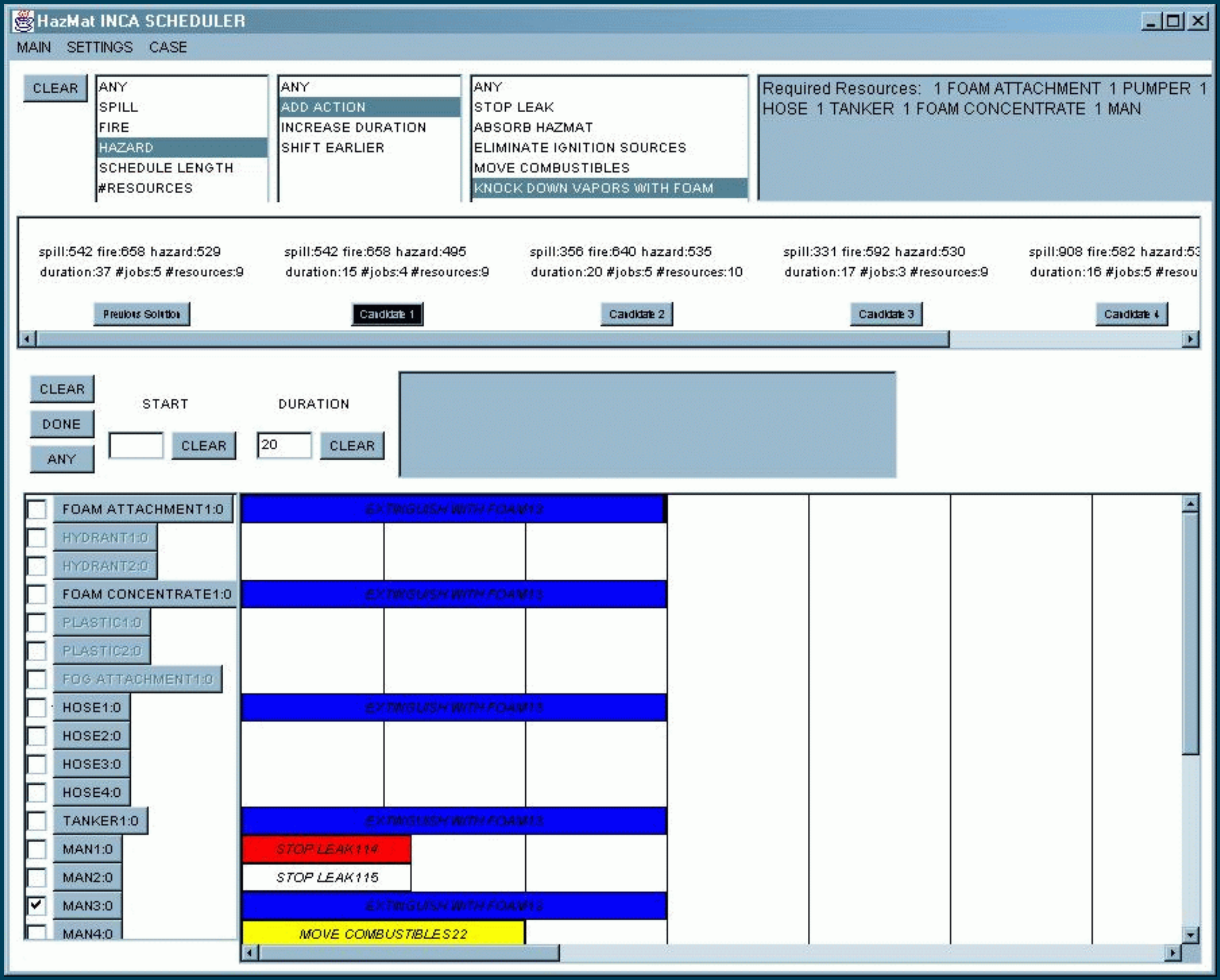
The INCA System

We implemented this design in INCA (Gervasio, Iba, & Langley, 1999), an interactive scheduler that:

- retrieves an initial schedule from a personalized case library
- suggests to the user improved schedules from which to select
- lets the user *direct* search to improve on certain dimensions
- collects user choices to refine its personalized utility function
- stores solutions in the case base to initialize future schedules
- invokes the user model to recommend future schedule repairs

As before, providing users with choices lets the system collect data on schedule preferences in an unobtrusive manner.

INCA: Interactive Scheduling



Experimental Results with INCA

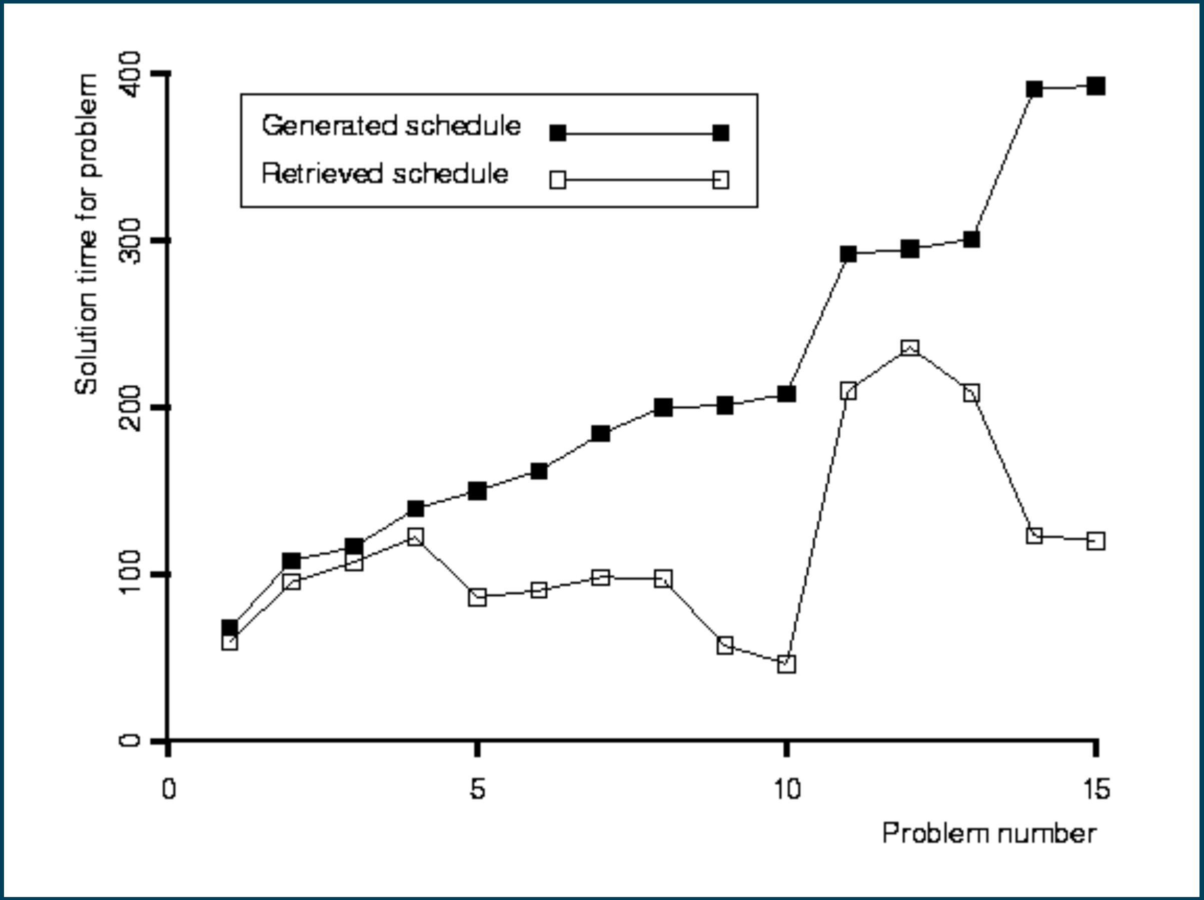
Experiments with the INCA scheduling system suggest that:

- it improves its ability to predict user choices over time
- personalized case libraries are more effective than generic
- its advice provides greater benefit on harder problems
- linear models give useful predictions even when false
- more detailed guidance speeds the user-modeling process

These studies (Gervasio et al., 1999) used a mixture of human and synthetic subjects.

INCA and Task Characteristics

Experiments with INCA suggest that retrieving personalized schedules helps users more as task difficulty increases.



Personalized Travel Advice

MyTravelAgent by MindShadow

Earlier Departure | Later Departure | Earlier Return | Later Return

Airline	Dep. date	Dep. time	Arrival	Ret. date	Ret. time	Ret. arr.	Cost	Duration	#Stops	#Connect	Layovers
American Airlines	Mar 13	6:35 AM	1:20 PM	Mar 18	8:03 AM	11:35 AM	560.00	10:17	2	2	2:26
American Airlines	Mar 14	6:35 AM	1:20 PM	Mar 19	8:03 AM	11:35 AM	476.00	10:17	2	2	2:26
American Airlines	Mar 13	7:59 AM	1:26 PM	Mar 19	8:05 AM	9:47 AM	545.00	7:09	0	0	0:00
American Airlines	Mar 14	6:33 AM	11:55 AM	Mar 19	8:05 AM	9:47 AM	524.00	7:04	0	0	0:00

OUTBOUND FLIGHT:

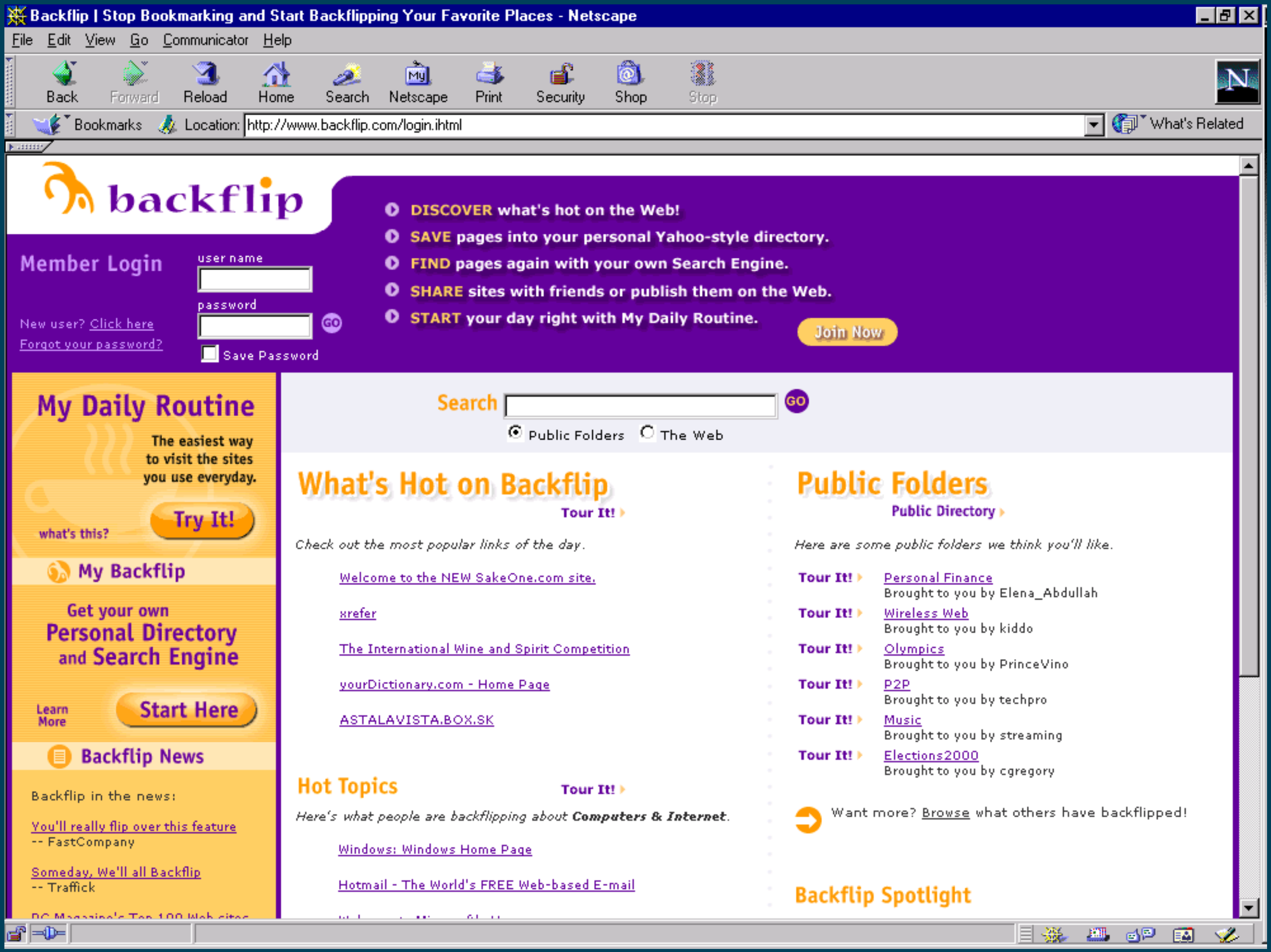
American Airlines flight 1228 on a McDonnell Dougl SP80
From: San Jose, CA (SJC) Wed, Mar 13, 2002 at 07:59 AM (PST)
To: Dallas/Ft Worth, TX (DFW) Wed, Mar 13, 2002 at 01:26 PM (CST)

RETURN FLIGHT:

American Airlines flight 639 on a Boeing 757
From: Dallas/Ft Worth, TX (DFW) Tue, Mar 19, 2002 at 08:05 AM (CST)
To: San Jose, CA (SJC) Tue, Mar 19, 2002 at 09:47 AM (PST)

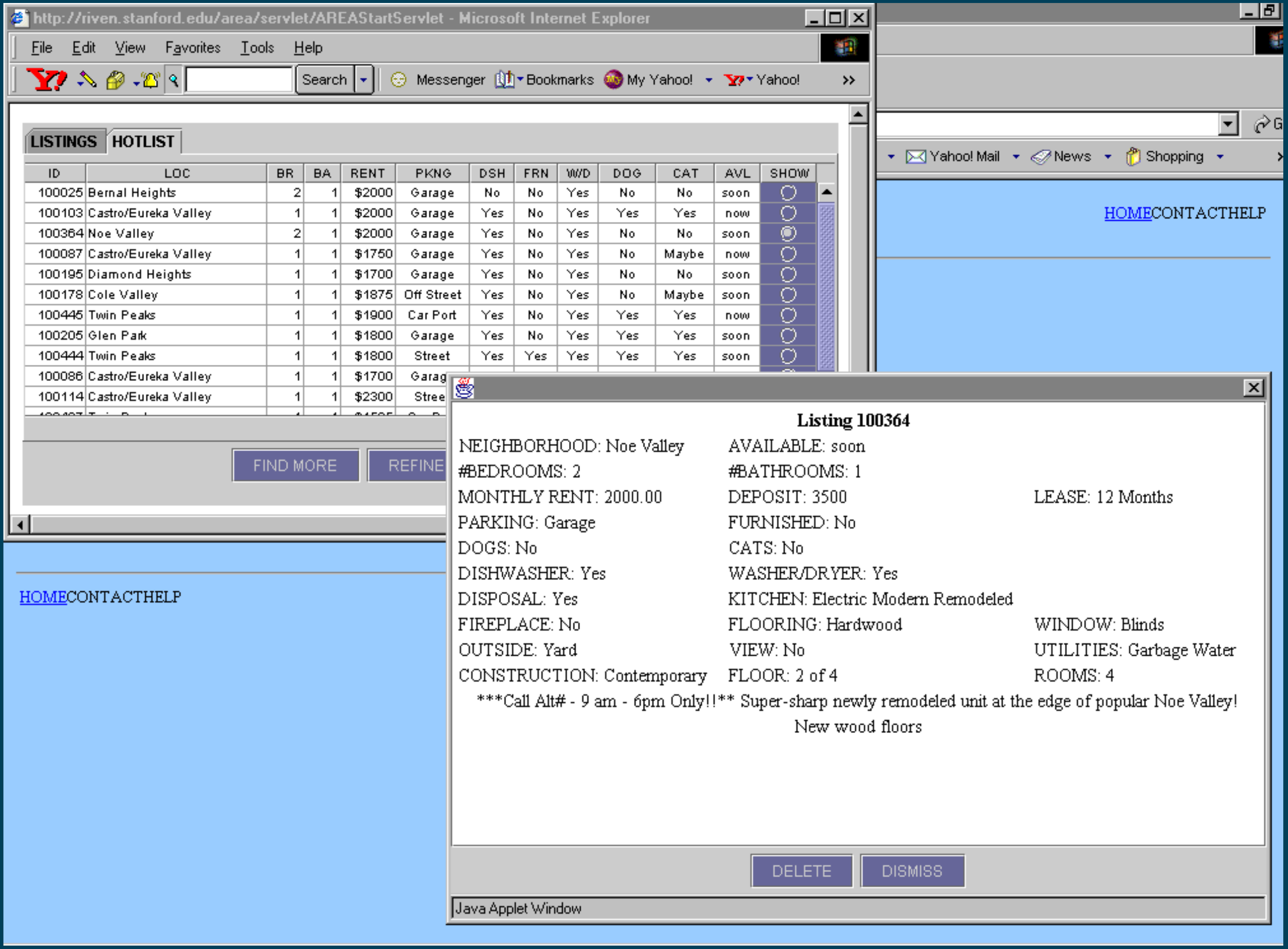
Select | Cancel

Personalized Bookmarks



Personalized Music Delivery

Personalized Apartment Finding

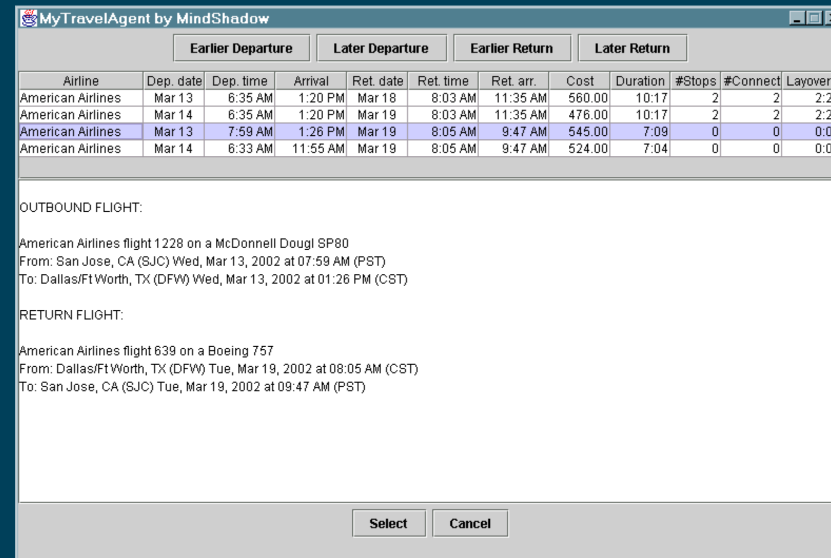


Discussion

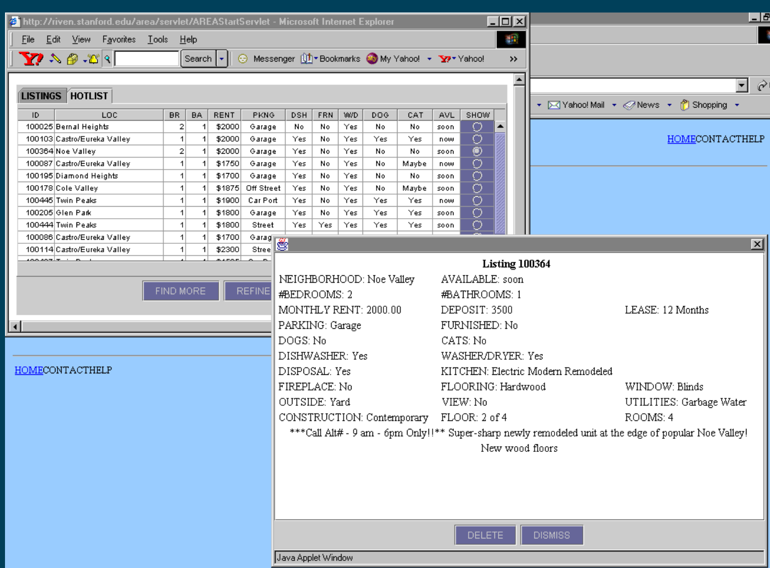
Presentation Styles

Radio / Sequential

Travel Aid / Tweaked Set



Apt. Finder / Ranked List



Stock Tracker / Classifier

Challenges of Adaptive Interfaces

Adaptive user interfaces have clear attractions but also pose some challenges to developers:

- formulation of user modeling as an induction task
- engineering of representation to support learning process
- unobtrusive collection of training data from users
- effective application of learned user model
- requirement for some form of *online* learning
- necessity for induction from *few* training cases

These challenges overlap with other applications of machine learning, but also raise some new issues.

The Promise of New Sensors

Adaptive interfaces rely on user traces to drive their modeling process, so they stand to benefit from developments like:

- GPS and cell phone locators
- robust software for speech recognition
- accurate eye and head trackers
- real-time video interpreters
- wearable body sensors (GSR, heart rate)
- portable brain-wave sensors

As such devices become more widespread, they will offer new sources of data and support new types of adaptive services.

Adaptive Interfaces as Psychological Models

We can view adaptive interfaces as automatically creating cognitive simulations, in that they:

- develop knowledge structures to describe user preferences
- make explicit predictions about the user’s future behavior
- explain individual differences through personalization

But we can distinguish two approaches to cognitive simulation:

- *process* models that embody architectural principles
- *content* models of behavior at the knowledge level

Both have roles to play, but content models are more relevant to personalization and adaptive interfaces.

Closing Remarks

In summary, adaptive interfaces integrate ideas from machine learning, intelligent agents, and human-computer interaction.

This approach to automated personalization of services offers:

- an alternative to the dominant "big data" perspective
- many unexplored niches for research and application
- challenges of system design rather than algorithm creation

These adaptive systems promise to change the way we interact with, and think about, computer software.