Optimal Bidding on Keyword Auctions

BRENDAN KITTS AND BENJAMIN LEBLANC

A b s t r a c t

Pay per click (PPC) auctions are used to sell positions in search engines. These auctions have gained increasing commercial importance, and many companies offer software to bid on these auctions. We present a trading agent for PPC auctions that is the first in our knowledge to use an explicit profit objective function. The agent creates a look-ahead plan of its desired bids, which allows it to exhibit intelligent behaviour including the ability to hold back money during expensive periods. We tested the agent in the latter part of 2003 in a live Overture auction. The agent generated four times the number of visits as human managers, in addition to reducing cost and variability.

Keywords: PPC, pay per click, exploration, case study, agent, search engine

OVERVIEW

In this paper we present a trading agent for pay per click (PPC) auctions. The agent exhibits many intelligent characteristics, including being able to plan ahead, allocate resources between auctions, and automatically explore. The first part of the paper will describe how PPC auctions work, including current automated approaches for bidding on these auctions. The second part will describe the major features of the agent including the formulation of its objective function, the method for estimating statistical functions and exploration strategy. The final part will describe an experiment in which we compared the performance of the agent against human controls.

AUCTION MECHANISM

A PPC auction is a continuous second-price auction for advertising space in search engines. For instance, if a user types in a search at Google, they will get back a set of listings. At the right hand side of the page there will be some sponsored listings. These sites have paid, on a PPC auction, to have their companies shown after the user has entered a search meeting their specification (see Figure 1).

This method of advertising is similar to the telephone Yellow Pages. In the Yellow Pages, advertisers pay more to have larger, more visible advertisements. In PPC, advertisers pay more to be listed higher in the returned results from an online search engine.

Each competitor enters a bid $b_{k,t}$ which is the amount they are willing to pay should a customer click on their advertisement in the search results for keyword k at time t. For instance, a company may be prepared to pay up to $11.03 for a customer who typed 'home loan California' and $5.02 for a customer who typed 'home loan'. In general, more specific terms are cheaper but generate less traffic (Figure 2).

The auctioneer — a search engine such as Google — sorts all of the bids that participants placed for their keyword. The auctioneer awards position 1 to the highest bid, position 2 to the second highest bid, and so on. Positions are re-calculated continuously throughout the day and participants may change their bids at any time. Participants may submit any bid over the auction minimum, but if the bid is too low they may find themselves placed at position 100 where they never get a click. Thus, there is a trade-off between keeping a bid low enough to maintain profitability and high enough to generate volume.

After each position is awarded, the auctioneer determines the final price that each competitor should pay. This amount is usually equal to the

A u t h o r s

Brendan Kitts
(bkitts@excite.com) holds the BInf, MA and MCogSc degrees. Brendan was a Lecturer in Computer Science at the University of Technology Sydney, and has headed scientific groups in both large and small companies. Brendan is currently working on large scale optimization problems for iProspect.
Benjamin LeBlanc
(bj_leblanc@hotmail.com) holds a BS in Computer Science and Geographic Information Systems. Currently his focus is on developing distributed system architectures to support large numbers of real-time non-homogeneous auctions. Prior to joining iProspect he was responsible for designing and developing software for airborne digital image acquisition.

Volume 14 (3): 186–201. www.electronicmarkets.org DOI: 10.1080/10196780420000245119

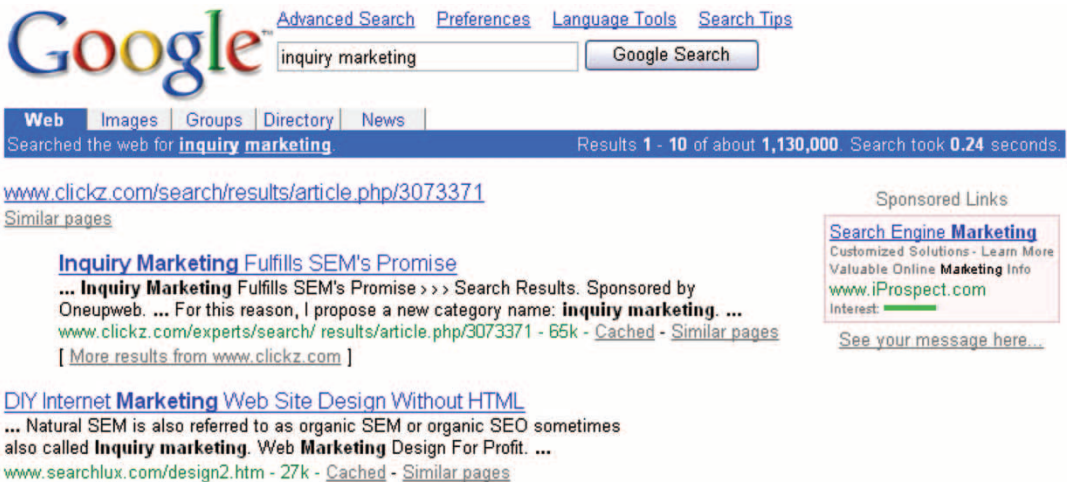

Figure 1. How PPC works. A user types in a query "inquiry marketing". In response, Google displays both natural results (vertically down the page) and sponsored results (small box at right)

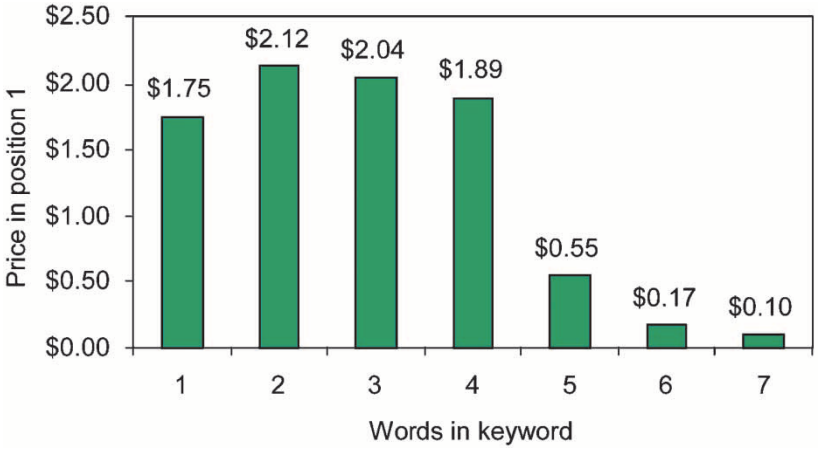

Figure 2. Price for position 1 versus the number of words in the keyword. For example 'college degree' is a 2-word keyword and would have an expected cost of $2.12 per click for position 1. The above data were compiled by scanning 410 Overture auctions on 6 November 2003

bid of the next competitor immediately below. Thus the auction is a second price auction. In terms of price quotes, Overture uses an open bid model where the prices for each position are visible to all on the auction. Google uses a sealed bid model. The major features of the PPC auction are summarized in Table 1 (Wurman *et al.* 2001, 2002).

In November 2003 approximately 25 companies hosted PPC auctions including, in order of size, Overture, Google and FindWhat. The largest auction company, Overture, generates $253 million dollars per quarter in revenue and is profitable (Overture Services Inc. SEC Filing September 2003). For comparison, eBay generates $531 million in revenue per quarter (eBay Inc. SEC Filing September 2003). More than just being big, PPC auctions are growing. Spending on Overture auctions has grown at a rate of 20% per quarter from 2001 to 2003.

Previous work on PPC auctions

Several companies market bid management software for PPC auctions. Companies such as KeywordMax, SureHits, DidIt and GoToast offer rule-based systems in which you can set minimum prices, maximum prices, desired positions, and so on. Here is a sampling of rules provided by GoToast (www.gotoast.com).

- *Timed position*: Set a position that you would like to maintain in the auction between certain hours of the day.
- *Relative positions*: Allows your listing to always be a certain number of positions above a particular competitor. Figure 3 shows an example of a bidding exchange that was probably caused by a relative to position rule.

Table 1. Comparison of the two leading PPC auction models using the conceptual framework proposed by Wurman *et al.* (2001, 2002)

Category	*Overture*	*Google*
Bidding rules		
Bidding rules	Any bid above minimum price will be accepted	Any bid above minimum price will be accepted
Information revelation		
Price quotes	Visible	Sealed
Quote timing	Quotes updated at specified frequency	Quote only updated after a new customer query
Clearing policy		
Matching function	Sort bids, award top positions to top bids, de-list bidders with poor performance	Sort bids, award top positions to top bids and apply position penalty or de-list for poor performance
Price	Second price	Second price
Tie breaking	Random	Random
Auctioneer fees	Per click	Per click
Closing conditions	None — Perpetual auction	None — Perpetual auction

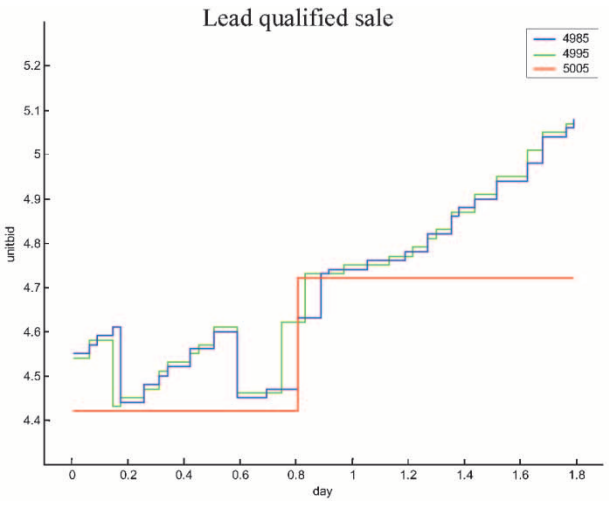

Figure 3. Bidding exchange detected in the fall of 2003 on the auction for 'Lead qualified sale'. The two active competitors are eLeadz and Microsoft. A relative to position rule may have been in effect, allowing one competitor to always appear in front of the other competitor

- *GapJammer*: Moves a bid price to one cent below your competitor's listing. Because auctions are second price, this forces the competitor above to pay the maximum amount for their position. This is an example of an *anti-social* bidding rule (Sandholm 2000; Brandt and Weiß 2001a, 2001b).
- *Move to gap*: Move to a gap greater than D in dollars. A 'gap' in an auction exists when there is a group of competitors who are d cents away from each other, followed by a steep rise ($D >> d$) to another group of competitors (see Figure 4 for an example). Say the auction prices for positions 4 to 1 are \$1.41, \$1.42, \$1.43, \$3.09. A person who bids \$1.44 would place themselves into position 2 for almost the same price as the person in position 4 (\$1.41). The region \$1.44 to \$3.09 is known as a 'gap'. Heuristically it can be a fairly effective strategy to look for these gaps and situate into them.
- *Move if cheap*: Execute further purchases of positions if the change in cost is less than x.

Rules are useful for catching exceptions and implementing business logic on branded terms. For example, President George W. Bush could create a 'keep GeorgeBush.com in position 1' rule to prevent the 'George Bush Talking Action Figure' becoming the top result in Google (as of the time of writing the action figure is in fifth position at \$0.21 per click http://www.toypresidents.com/view_product.asp?pid=1).

However, there are limitations. If there are 1,000 keywords, then one would need to create 1,000 rules to manage them. There is no guidance on how to come up with these rules, and maintaining and adjusting the rules amidst a continuous, non-stationary auction can be tedious.

A recent step forward in the field of automated PPC bidding has been the development of a rule that we will term a 'preset'.[1] These rules allow the user to type in a desired goal for each keyword — for instance a return per dollar spent — and the bidding agent automatically adjusts the bid price so as to meet this goal (Pasternack 2002, 2003). An example of how an ROAS rule works is as follows. Let r_k be the expected revenue conditional upon a click for keyword k calculated using a weighted average of historical data. Given a desired R_k* that the client wants to achieve, and assuming a stationary r_k we can calculate a bid b_k* to achieve this ROAS as follows

$$b_k^* = \frac{r_k}{R_k^*}$$

Unfortunately, these rules have many shortfalls. In order to avoid setting the price of new keywords to zero, a certain number of clicks usually needs to be collected before the rule can be activated. The rule relies upon the user to fill in a sensible ROAS. For example, if the ROAS preset is set too high (for example, because we want the highest return, so we set the ROAS to \$10 per \$1 spent),

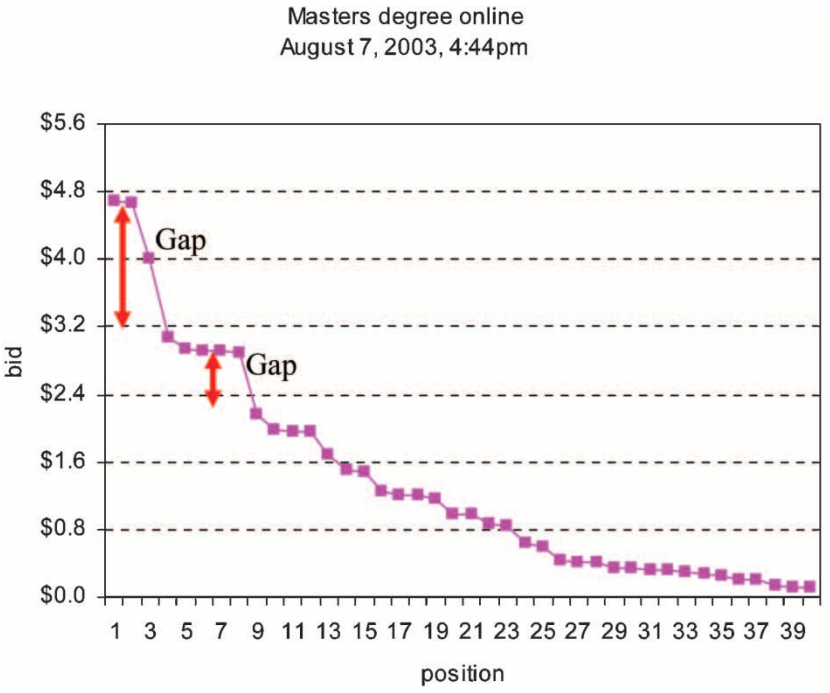

Figure 4. Price gaps in a typical open auction. The tendency of competitors to bid one cent over each other results in clusters of bidders, followed by a large jump in price or 'gap' to another cluster

the user will generate a high margin, but low volume. If the ROAS is too low (for example, $1.05 per $1 spent), the user will generate high volume but low margin. Somewhere between these two extremes there is an optimum for profit, but the rule offers no insight into how to achieve the optimum.

One could even imagine developing a procedure to find the optimal profit bid for each keyword. However, we then face the problem of allocating resources *between* keywords. For example, a keyword may achieve its ROAS only by exhausting all available funds, leaving nothing for other keywords that might have been able to generate higher returns of profit.

These techniques have so far failed to solve the core business problem — which is how to generate the largest profit under a finite budget. To solve this problem we will re-express the PPC auction problem as a large-scale optimization problem.

FORMAL STATEMENT OF THE PROBLEM

We have K keywords for which we want to place bids in auctions. The present time is t_r. We have divided the future into T discrete time units for the purposes of developing a bidding plan for each future time unit. Although T is finite, it could extend arbitrarily far into the future, and we will refer to the number of hours in the future as the *planning horizon* of the system.

For each bid $b_{k,t}$, that we place for a keyword k, at a given time t, the search engine company will compare our bid with those of other competitors on the auction, and award us a position according to the unknown function $p(k,t,b_{k,t})$. Given placement into a position, we can also expect to register a certain number of clicks per hour from customers who are searching on our targeted keyword term, see the advertisement in their search results, and click on it, $c(k,t,p(k,t,b_{k,t}))$. From these clicks some number of customers may generate an order, and the average revenue generated from each click on k can be given as r_k.

We desire to find a price $b_{k,t}$ to assign to each keyword, at each future time. This vector of prices, called the *bid allocation plan*, should have the property of maximizing the summed profit during the planning horizon, while not exceeding the maximum summed expenditure or 'budget' that we are allowed per day.

A set of rules may also be specified to exert finer control over the behaviour of certain keywords. Rules are discussed more fully in the following section, where we present a simple algorithm to translate keyword level constraints into $BMIN_{k,t}$, $BMAX_{k,t}$ constraints that can be used in optimization.

The auction prices that each competitor is paying for each position on auction k, and at time t, will be defined as $\{b_{k,t}^{(1)}, b_{k,t}^{(2)}, \ldots, b_{k,t}^{(Pk)}\}$ where P_k are the number of competitors on auction k. We may capture any position $p \in [1..P_k]$ by setting our bid to just above competitor p's bidding price, $b_{k,t} = b_{k,t}^{(p)} + o$ where o is the minimum bid increment on the auction (usually one cent). If we want to be in the lowest position we can bid the official minimum bid O (in Overture this is $0.10). On sealed

auctions, the prices of each position, $b_{k,t}(i)$ are not disclosed, but may be discovered by exploring the auction with different bids to infer competitor price points.

Formally, our problem is to find

$$b_{1,1,..,K,T} : \max \sum_k \sum_t \pi_{k,t}$$
$$\pi_{k,t} = (r_k - b_{k,t}) \cdot c(k,t,p(k,t,b_{k,t}))$$

subject to

$$\sum_k \sum_t b_{k,t} \cdot c(k,t,p(k,t,b_{k,t})) \le Budget$$
$$0 < O \le BMIN_{k,t} \le b_{k,t} \le BMAX_{k,t}$$
$$b_{k,t} \in \{O,\ b_{k,t}^{(Pk)} + o,\ b_{k,t}^{(Pk-1)} + o,\ \ldots,\ b_{k,t}^{(1)} + o\}$$

The problem can now be solved as an integer programming problem. However, there are several unknowns; namely r_k , $c(k,t,p)$, $p(k,t,b)$, $b_{k,t}^{(i)}$, $BMIN_{k,t}$ and $BMAX_{k,t}$. We describe our method for estimating these functions in the following sections.

RULES

Rules are implemented by translating behavioral specifications into optimization constraints. For example, position minimum, position maximum, bid minimum and bid maximum rules can be translated into $BMIN_{k,t}$ and $BMAX_{k,t}$ constraints that are used to bound the optimization. This is described in more detail below.

Bid minimum and maximum, Position minimum and position maximum

Let $bidmin_{k,t}$ and $bidmax_{k,t}$ provide the user-specified minimum and maximum bids allowed for a keyword at a particular time. Let $posmin_{k,t}$ and $posmax_{k,t}$ provide the user-specified lowest and highest positions allowed for a keyword at a particular time. We can use these rules to constrain the optimizer to, for instance, 'stay between positions 2 and 10 and prices \$1.00 and \$10.00 and generate the maximum profit'. It is possible that a combination of these rules could be infeasible. For example, if the user specified 'stay in the top 3 positions, and spend at most \$0.10', on an auction where the top three positions were (\$5.10, \$5.09, \$5.00), there would be no bidding positions that would satisfy the user's rule. We convert these four possibly inconsistent constraints into two feasible constraints $BMIN_{k,t}$ and $BMAX_{k,t}$ using the algorithm below. The algorithm ensures that bids will never exceed $bidmax_{k,t}$, and that prices meeting the position criteria will be selected if possible. If no prices are feasible, $bidmin_{k,t}$ will be relaxed to ensure feasibility (see Figure 5 for an example).

$$BMAX_{k,t} = o + \min\{bidmax_{k,t} - o,\ b_{k,t,posmax}\}$$
$$BMIN_{k,t} = o + \max\{bidmin_{k,t} - o,\ b_{k,t,posmin}\}$$
$$\quad , \text{if } \max\{bidmin_{k,t} - o,\ b_{k,t,posmin}\} \le BMAX_{k,t}$$
$$= o + \max b_{k,t}^{(p)} : b_{k,t}^{(p)} \le BMAX_{k,t}$$
$$\quad , \text{otherwise}$$

where

$$b_{k,t,posmin} = \min b_{k,t}^{(p)} : posmin_{k,t} \le p \le posmax_{k,t}$$
$$b_{k,t,posmax} = \max b_{k,t}^{(p)} : posmin_{k,t} \le p \le posmax_{k,t}$$

ESTIMATION OF UNKNOWN FUNCTIONS

Clicks estimation

The clicks model, $c(k,t,p)$, estimates the number of user clicks per hour generated in each discrete time-auction-position. This function could be solved by creating functions mapping position to clicks for each discrete keyword-time in the planning horizon. However, a lot of samples would be required to adequately cover each keyword-time. Each sample must be bought with real money.

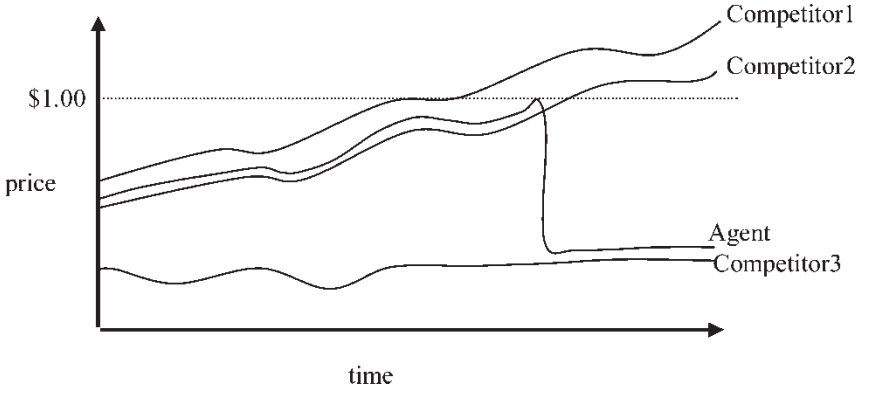

Figure 5. An example of how position and price rules interact. We defined $posmax_k = 2$ and $posmin_k = 2$, which is equivalent of saying 'stay in position 2'. We also defined a $bidmax_k$ of \$1.00. The agent maintains position 2 until the price it would pay exceeds \$1.00. At that time, following the feasibility constraint logic, it drops to position 3.

We have improved the speed and cost associated with model induction as follows. If we don't have any data for a particular hour, we draw data from surrounding hours in order to predict the demand for the hour in question. The data taken from the surrounding hours is applied a weight w_i between 0 and 1 that is proportional to the time similarity, giving rise to a Weighted Least Squares problem (Carroll and Ruppert 1988, Ryan 1997).

Therefore instead of having to sample each discrete time t, we can draw data from partially sampled nearby hours, but where those hours are de-weighted accordingly (Kalton and Kasprzyk 1986, Specht 1991). The demand at each keyword k time t position $p_{k,t}$ is subsequently modelled as an exponential function:

$$c(k,t,p_{k,t}) = \theta_k e^{\omega_k p_{k,t}}$$

θ_k and ω_k are shape parameters selected so as to minimize the fit between the observed and predicted clicks for a given keyword, time, position

$$\sum_{i:k_i=k}^{N} w_i \left[c(k,t_i,p_i) - c_i\right]^2$$

i represents a time-bid-position-click observation of keyword k with time t_i, position p_i and clicks c_i. w_i is computed as:

$$w_i = \prod_u \frac{1}{1+e^{-s(t_i,t,u)\alpha_u - \beta_u}}$$

where s is the time separation in units u of hours, days, or weeks between the historical time t_i and the forecast time t, and α_u , β_u are kernel parameters. Figure 6 shows the kernel functions, Figure 7 shows an example clicks function, and Figure 8 shows how the clicks function varies through time.

Market estimation

Market estimation consists of estimating the unknown function $p(k, t, b_{k,t})$, a function that predicts the position we would receive if we were to submit a price for a keyword at a particular time. Estimation of p is fundamentally a *price prediction* problem and has been discussed in the context of other auctions (Saroop and Bagchi 2002, Schapire *et al.* 2002, Wellman *et al.* 2003b, 2004). In its most extreme formulation, price prediction may encompass predicting the behaviour of individual competitor agents.

Open bid. Open bid auctions such as those run by Overture provide the price for every position on the auction at

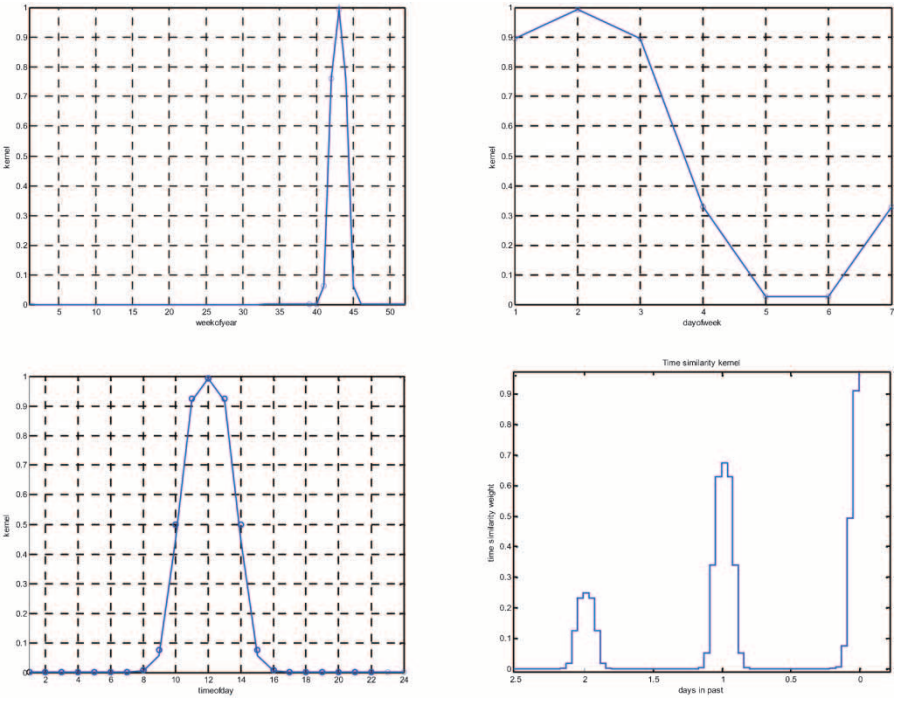

Figure 6. The time similarity weighting function is the product of three kernels, one for hour similarity, one for day similarity, and one for week similarity. (*top left*) Week similarity, (*top right*) Day similarity, (*bottom left*) Hour similarity. (*bottom right*) shows a close-up of the resulting time similarity function across a trailing two and a half days. Time 0 on the graph is Tuesday 30 October 10:44am. Tuesday 9:44am is weighted at 0.9, Tuesday 8:44am is weighted at 0.5, and Tuesday 7:44am is weighted at 0.08. Monday at 10:44am is weighted at 0.7. Sunday 10:44am has a weight of 0.25

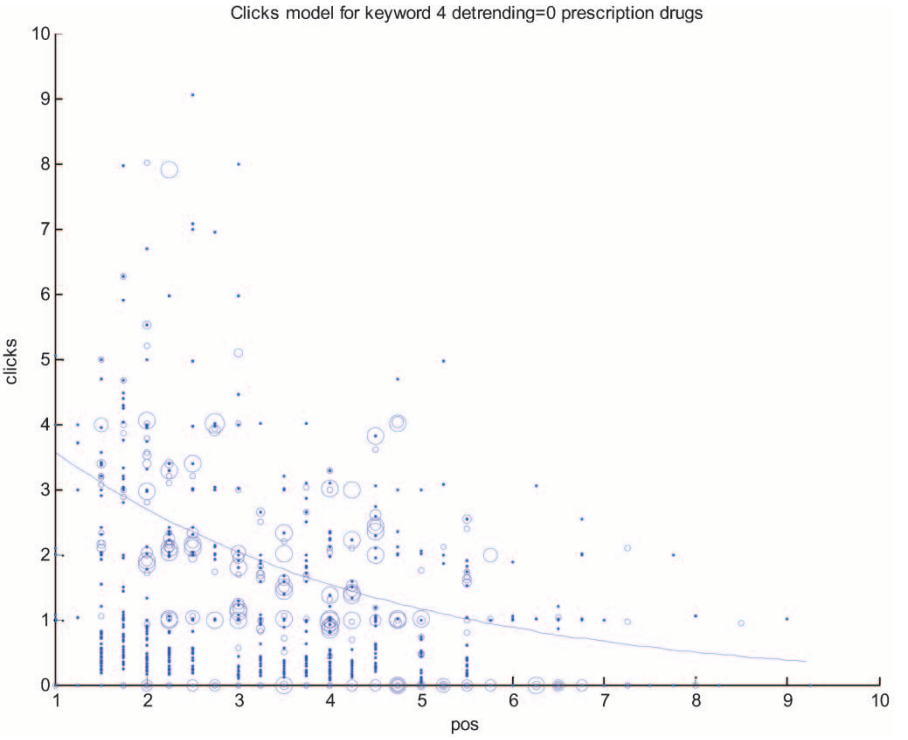

Figure 7. Clicks function for auction for 'Prescription drugs' at time *t*. The function predicts the number of clicks per hour that will be generated at each position in the auction, for keyword *k* at time *t*. Each circle is an actual observation of position-clicks for this keyword, with the similarity between the time of historical observation and *t* represented as the size of the circle. These weightings in effect 'carve out' a set of data-points that are likely to be representative for this time, amidst a large number of relevant and irrelevant cases. The function shown above was from a sealed auction in which positions were reported as averages rounded to the nearest 1/4

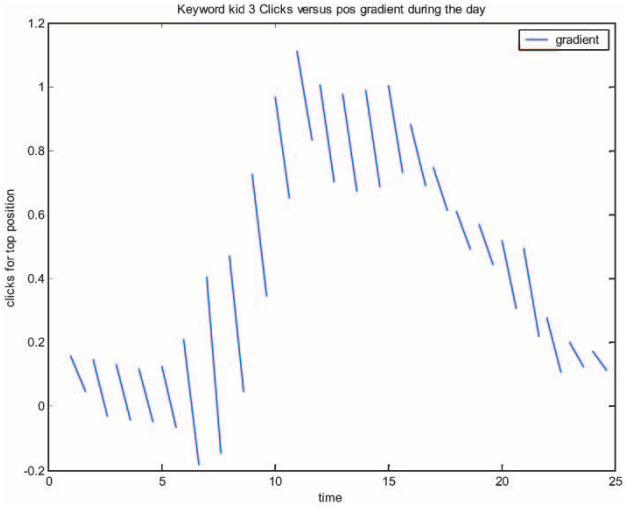

Figure 8. Clicks and derivative of clicks with respect to position at position 1 for an example auction, shown over a future 24 hour period. The number of clicks at position 1 is represented as the left-most part of the line, and the gradient at that point as the slope of the line. This shows how each $c(k,t,p)$ function can vary according to time of day. The different $c(k,t,p)$ functions allow the agent to alter bidding during expensive versus cheap periods of the day, a practice known as "day parting"

the present time. Therefore $p(k, t_r, b_{k,t_r})$ for the present time t_r is given. The only remaining unknown is the price of those positions in the future $t > t_r$. An auto-regressive neural net can be used to perform this prediction. The method below creates a separate model for each position p, predicts the differenced change in price from one time-unit to the next $y_{i,\tau}^{(p)}$ at τ time-units in the future, based on increments in price x_i observed in the last L time-units, and uses a normalized Radial Basis Function to accomplish the fit (Girosi and Poggio 1993, Masters 1995, Karur and Ramachandran 1995). For example, if two agents were having a bidding war in position 5 and 6, the agent would detect the series of increases, and would forecast an elevated future price. In the formula below $x_{1..p}$, are a randomly chosen subset of cases from x_i which act as basis vectors and represent a prototypical formation of price increments.

$$p(k,t,b_{k,t_r+\tau}^{(p)}) = p$$

$$b_{k,t_r+\tau}^{(p)} = b_{k,t_r}^{(p)} + \sum_{j=1}^{\tau} y_{i,j}^{(p)}$$

$$y_{i,\tau}^{(p)} = \sum_{v} a_{v,\tau} \frac{f(|x_v - x_i|)}{\sum_{v} f(|x_v - x_i|)} + a_{0,\tau}$$

where $f(z)=z^2+\ln(z+1)$

$$x_i = (b_{k,t_r-L+1}^{(p)} - b_{k,t_r-L}^{(p)}, b_{k,t_r-L+2}^{(p)} - b_{k,t_r-L+1}^{(p)}, \ldots, b_{k,t_r}^{(p)} - b_{k,t_r-1}^{(p)})$$

and $a_{y,t}$ are chosen so as to minimize

$$\sum_{i:k_i=k} \sum_{\tau} \left[y_{i,\tau}^{(p)} - (b_{k,t_i+\tau}^{(p)} - b_{k,t_i+\tau-1}^{(p)}) \right]^2$$

Sealed bid. Sealed bid auctions such as those run by Google are more difficult. Sealed auctions inform you as to the position you have been assigned, but not the position of any of the other competitors. As a result, if you are in position 5, it is impossible to know exactly how much a position of 1, 2, 3 or 20 costs, or even if there are 20 positions in the auction.

To build the model for $p(k,t,b)$ on sealed auctions, we will use the agent's history of bids and positions as datapoints to infer the outcome of submitting a particular bid. We use an exponential function to weight these market samples, so that observations that were taken further in the past receive less weight. Our exponential weighting is given by:

$$g_i = m^{t-t_i}$$

where m is a number between 0 and 1, and $t-t_i$ is the difference between the time being forecasted and the time of the historical record expressed in hours. An exponential model is chosen to model the function mapping bid to position as follows:

$$p(k,t,b_{k,t}) = \begin{cases} \psi_k e^{\xi_k b_{k,t}}, 0 \le b_{k,t} \le b_{k,t}^{(1)} \\ \psi_k e^{\xi_k b_{k,t}^{(1)}}, b_{k,t} \ge b_{k,t}^{(1)} \end{cases}$$

where ψ_k and ξ_k are exponential shape parameters and $p(k,t,b_{k,t}^{(1)}) = 1$. Parameters ψ_k and ξ_k are selected to minimize

$$\sum_i g_i \left(p(k,t,b_i) - p_i\right)^2$$

Figures 9 and 10 show examples of this model.

Revenue estimation

Revenue conditional upon clickthrough is one of the most difficult variables to estimate because of the rarity of sales events. In modelling this variable, we explicitly assume that revenue per click is formally independent of position and time. In other words, if a customer manages to click through to the site, their probability of

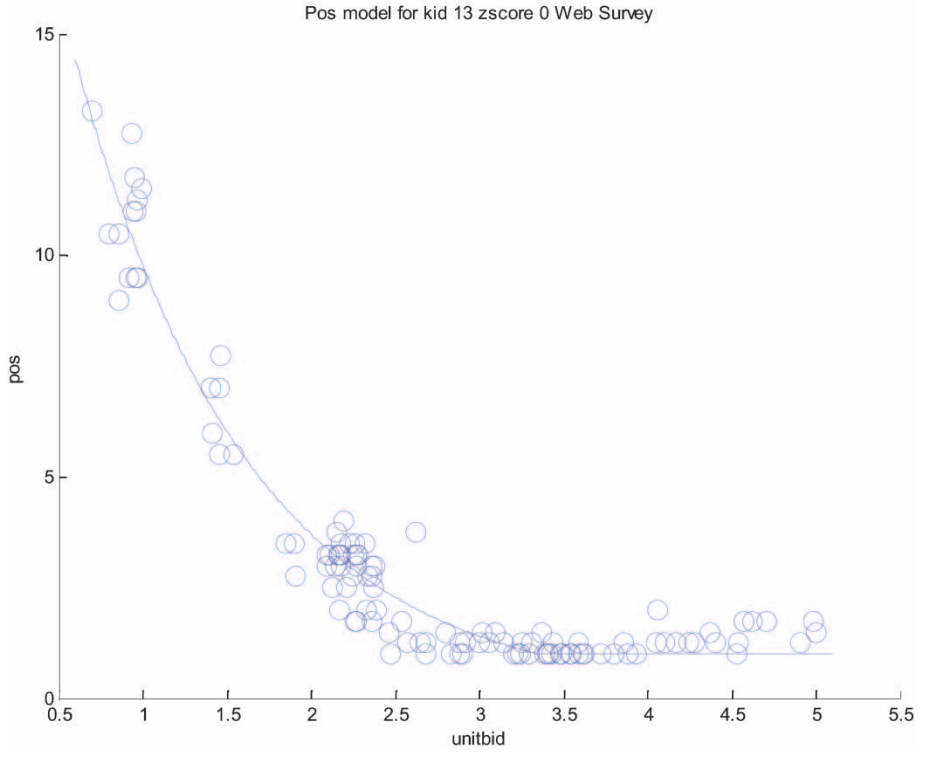

Figure 9. Position function for 'web survey'. Each circle represents an observation of time-bid-position. The size of each circle represents the weight g_i of this observation; but since $m = 1.0$ all observations have equal weight. Above a bid of $3.20, the position awarded is 1 regardless of price

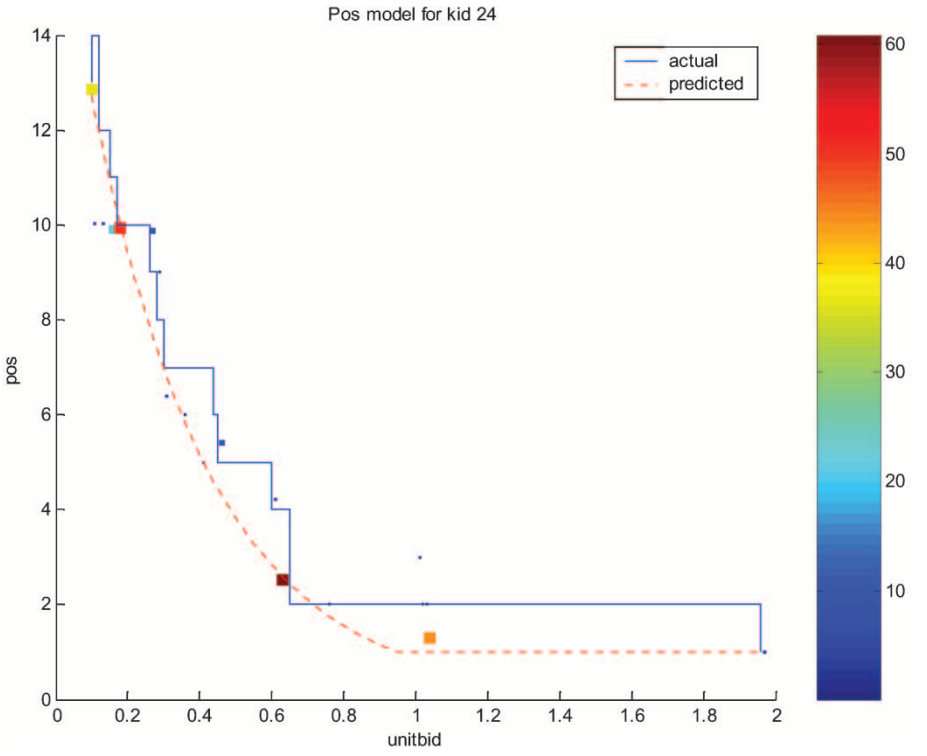

Figure 10. Example of market estimation. The filled squares are price-points that the agent tested in the past, with their size equalling their case weight g_i. The dashed line is the p model for this auction, based on the historical samples. The solid line is the true state of the auction. The prediction based on historical data closely matches the actual auction

generating revenue, and the amount they generate, will be the same after arriving, regardless of the time of their click (e.g., midnight versus midday), or the position from which they clicked (e.g., 2 versus 20). This assumption was verified using a Chi-Square test for independence, which indicated that the empirical distribution of revenue per click with position and time was highly probable under the null hypothesis. This assumption allowed us to model this variable as a simple average, rather than a function of variables. Revenue per click, r_k, is calculated as follows:

$$r_k = \frac{\sum_i h_i r_{k,i}}{\sum_i h_i c_{k,i}}$$

where $\sum_i h_i = 1$.

Figure 11 and 12 show the agent's ability to forecast traffic based on bidding decisions, and to correctly estimate demand patterns by hour and by position.

MODEL DEGENERACY

We now have a set of keyword-time models. However, what happens if something goes wrong? Montgomery and Rossi (1999) reflect our own experience by noting that consumer data often show contradictory and nonsense relationships due to uncaptured externalities. A typical example is a price — demand curve showing increasing demand for increasing price. We must autonomously overcome these problems for thousands of models every hour.

After each market and clicks model is created, it is tested for quality in two ways. First the model is examined to see if it violates any economic assumptions. These economic criteria include the following tests:

- clicks at position 1 are greater than zero, $c(k,t,1) > 0$
- position for bid of $0.00 is greater than or equal to 1, $p(k,t,0) \geq 1$
- the derivative of clicks with respect to position is negative $\frac{\partial c}{dp} < 0$; and
- derivative of position with respect to bid in the range $P_k \geq b > 0$ is negative $\frac{\partial p}{db} < 0$.

The second way we test model quality is by testing prediction accuracy on a small percentage of hold-out cases. The model is required to show an average percentage error below a preset threshold. If the model fails either of these tests, it is defined as 'degenerate'. If a model for keyword k time t is degenerate, we need to modify the processing of this keyword-time in several ways.

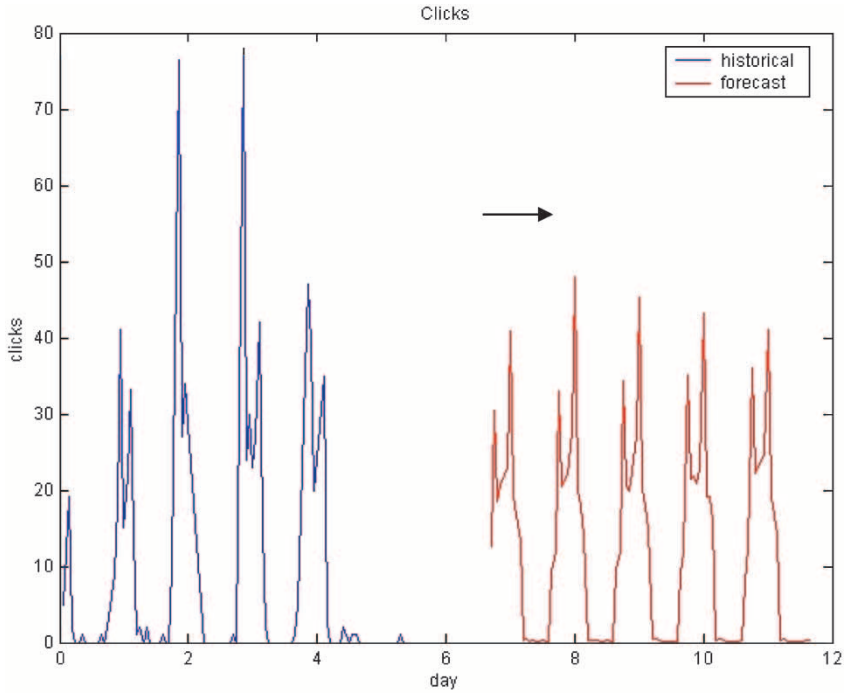

Figure 11. Clicks generated each hour for a live auction managed by the agent. The arrow indicates where the forecasts begin. This particular client implemented shut downs of bidding on nights and weekends, which involved dropping the bid price to $0.10 on Saturday, Sunday and nightly from 9pm to 7am. These were implemented as timed $bidmax_{k,t}$ rules, and the agent shows the shut-downs in its forecast

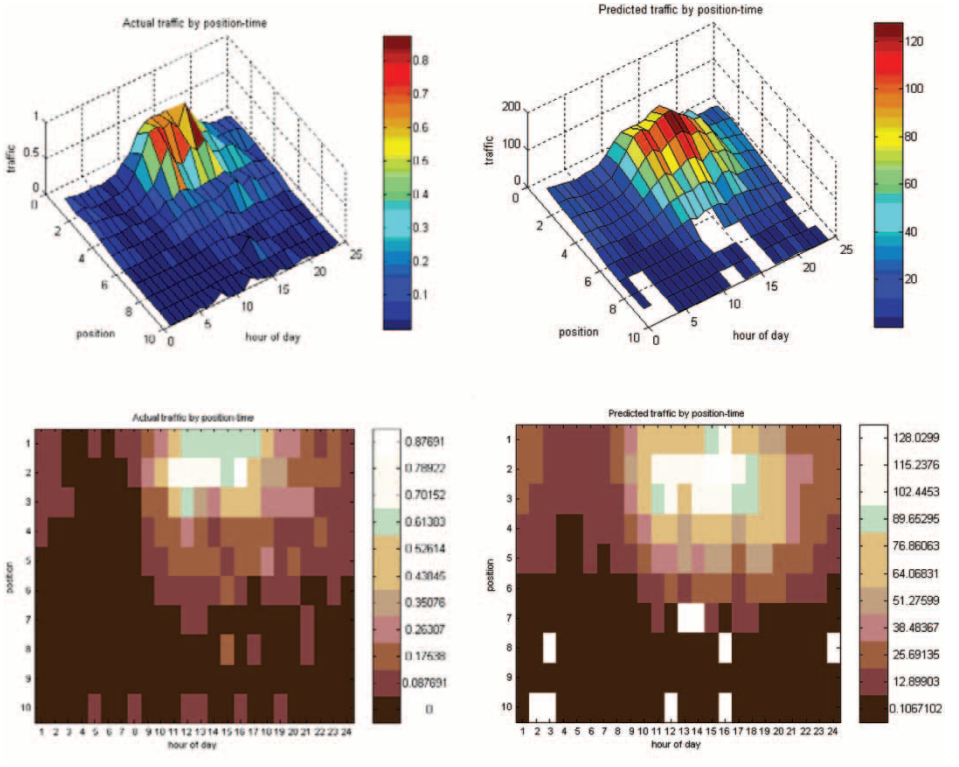

Figure 12. Clicks function by position and time. (*Left*) Actual measured clicks per keyword per hour over the history to-date. (*Right*) Predicted clicks per hour. Both surfaces are aggregated over all keywords. Missing sections in the predicted graphs are areas where a forecast was not recorded because of a constraint violation. The predicted clicks closely match the actual clicks, suggesting that the underlying keyword models are good. Predicted graphs show sum of clicks, actual graphs show average clicks per keyword

Bid determination

Once we designate a model as degenerate, the bid determination changes dramatically. Rather than search for an optimum bid, the agent now tries to insert a bid in an attempt to *repair* the model. The agent does this by initiating an exploration of the surrounding bidding landscape, so as to obtain more samples and fix the errant model. This is implemented by switching on a keyword-level exploration rule for this keyword-time. The exploration algorithm is described forthwith.

Budget

Although the keyword has escaped our ability to model it, it will still be a drain on the budget. Costs are estimated using mean imputation if none of the keywords' times were successfully modelled. If some of the keyword-times were successfully modelled, we assume a trend in time to fill in omitted data points (Little and Rubin 1987, Schafer 1997, 1999).

EXPLORATION STRATEGY

The exploration — exploitation tradeoff is one of the most resistant problems in machine learning (Holland 1975, Kaebling *et al.* 1996). The agent needs to be profitable and so must choose optimal bids. However, the auction is also continuous and non-stationary — competitors arrive and depart, changing the profitability of each position. Unlike other reinforcement learning problems, the non-stationarity of PPC auctions means that positions need to be constantly re-explored. We have tested both a random exploration method, that samples positions with a uniform random probability, as well as the Interval Estimation algorithm developed by Kaebling (1993), which selects positions so as to reduce the upper bound error on the models. Both methods perform well.

EXPERIMENTAL RESULTS

Design of experiment

Alpha Online University (not their real name) is an accredited online university that offers Bachelors, Masters and PhDs degrees in business and information technology. A PPC test program was initiated and involved nine words and a budget of $500 per month.

A human Marketing analyst managed the auction account from August until 14 September 2003 (period1). This individual would log onto the PPC auction manually and adjust bids. The individual who managed this account was employed to do the best job possible. From September 15 to October 30 (period2) we used our bid agent to autonomously place bids and optimize spending each hour.

Our objective was to optimize profit, but because the client initially did not have keyword profit data, we used the client's historical profit data in order to estimate the profit of each click. We estimated $2.97 per click as the value of a customer reaching the client's site. We ran our optimization based on this number.

After presenting results on the program's performance in late October, the client requested that we stop the experiment on October 30, and re-launch with an expansion in funding and keywords.

Results

Clicks approximately quadrupled after the agent came online, while expenditure decreased slightly from $18.40 to $16.60. The change in clicks was found to be statistically significant ($p < 0.01$) according to a Wilcoxon rank sum test. Table 2 shows the overall results. Figures 13 to 16 show timeseries.[2]

Analysis of agent strategy

Most clicks originated from two keywords: 'clep test' and 'learn java'. These two keywords were cheap — $0.20 and $0.93 per click respectively for position 1, and yet also generated a large number of clicks. In contrast, more popular keywords such as 'Education degree online' could cost as much as $5.50 for position 1 (Table 3). As a result, the agent learned that it could reduce its spending and generate a lot of clicks by moving to position 1 on the former two auctions.

Intelligent gap behaviour without rules

Recall that some companies have developed rules that move customers into gap positions based on user criteria (see Figure 4). After deploying the agent we noticed an interesting behaviour. The agent appeared to seek out and position itself in gaps. Perhaps it is obvious that this would have occurred, since there is a cost to clicks

Table 2. Experimental results for Alpha Online University

	Experimental results				
Metric	μ_1	μ_2	σ_1	σ_2	*p*-value [(a)]
Clicks	10.86	40.03	6.24	13.93	< 0.01
Expenditure	$18.41	$16.68	$10.12	$4.98	0.8586
CostPerClick	$1.74	$0.46	$0.42	$0.17	< 0.01

Note: [(a)] Wilcoxon sum rank test.

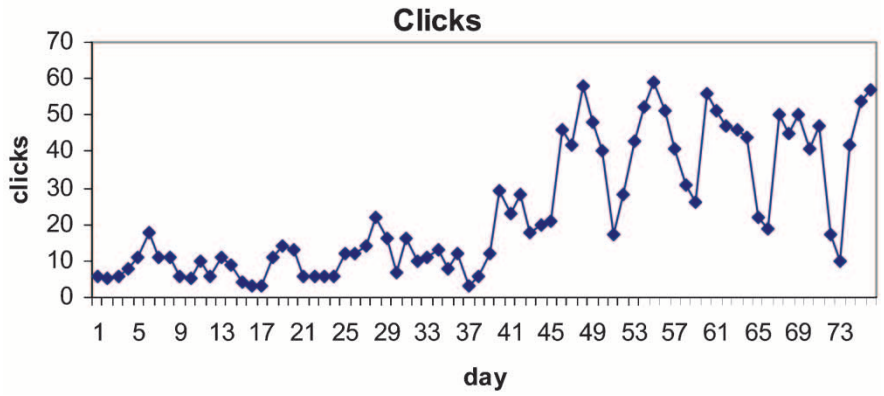

Figure 13. Clicks per day under human management and PPC agent management. Day 43 we brought the agent online. Clicks approximately quadrupled after the agent was brought online

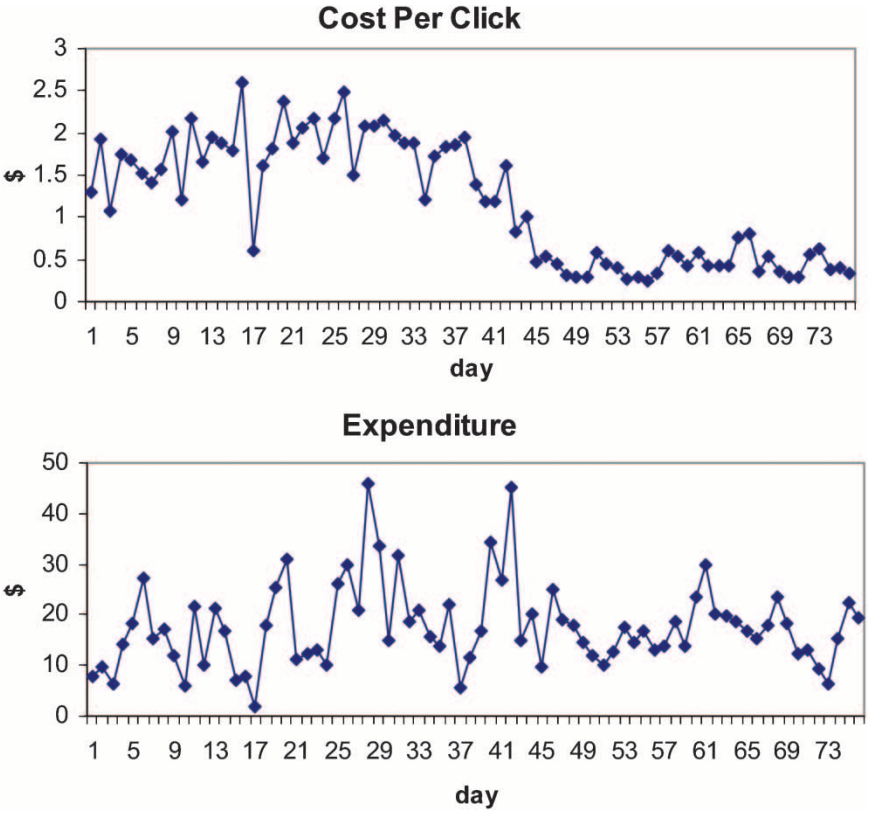

Figure 14. Cost per click and expenditure per day before and after PPC agent was brought online. The PPC agent was started on day 43 of this timeseries. (*top*) Cost per click dropped from around $1.50 per click to only $0.50 per click (*bottom*) Expenditure before and after the agent remained roughly the same. This was because the budget of $500 per month or $16 per day was still the same before and after. One noticeable change, however, is that the variance in spending is much lower after the agent came online

advantage in being in a gap. However, it was interesting to see. The gap finding behaviour is not reliant upon user-specified parameters. Rather, the agent constantly evaluates its utilities, and moves into gaps because the profit utility is better in those positions.

Figure 17 shows an example of this gap-finding behaviour in the auction during the experimental period. This type of behaviour was typical, and the positioning of the agent into gaps provided a convenient way for us to check that the agent was operating properly.

Expenditure behaviour

Target expenditure was $16 per day. When we look at actual expenditures in both periods, the actual

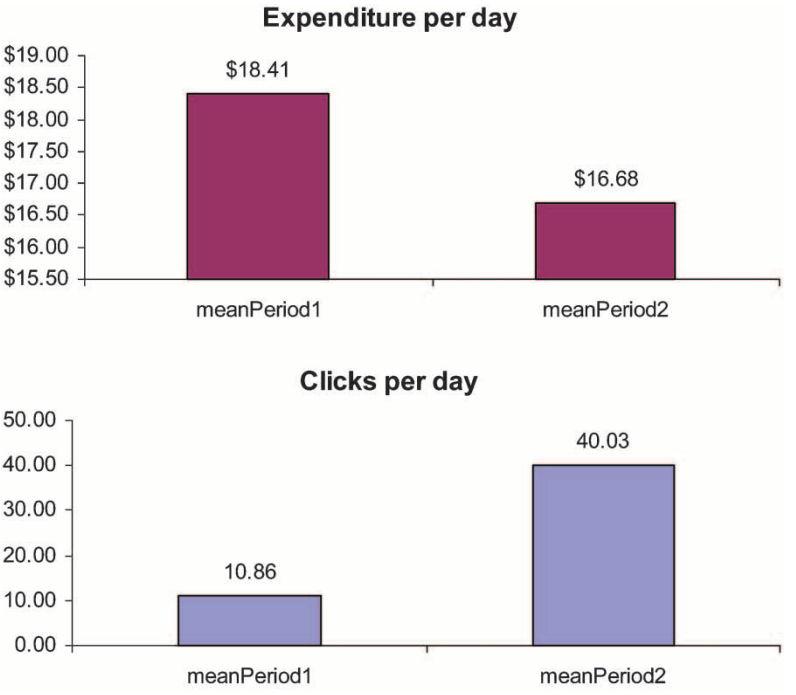

Figure 15. Clicks and expenditure per day during control and experimental period

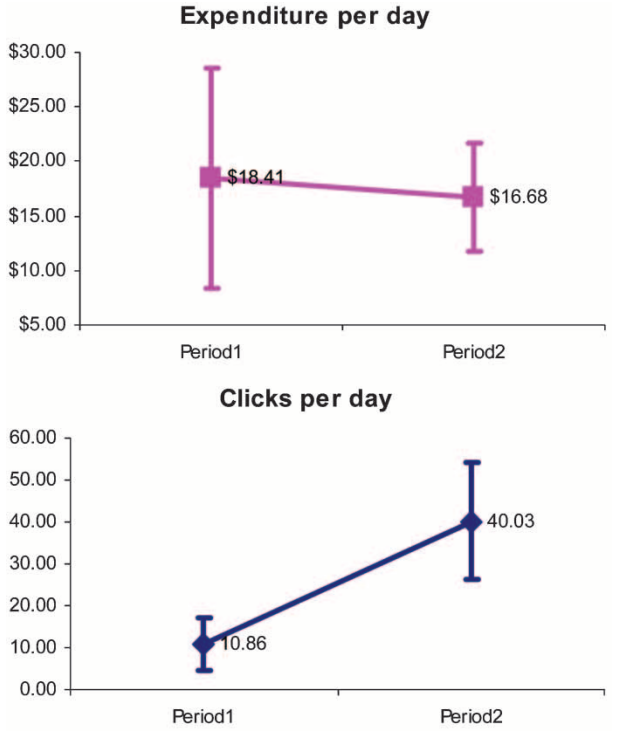

Figure 16. Clicks and expenditure per day during control and experimental period. Error bars show standard deviations. The increase in clicks is obviously large, however we also found some interesting results with expenditure. Expenditure in the experimental period is closer to the target of $16 per day ($18.41 to $16.68 per day). In addition, the standard deviation for expenditure *decreased* in the experimental period ($10 to $5). This suggests that agent management is not only effective in maximizing clicks per dollar spent, but it may also be effective in dampening out noisy increases and decreases in demand so as to meet its budget. We have termed the continuous trimming of spending as *feedback stabilization*

Table 3. Analysis of cost to click ratio for the keywords in our experiment

Keyword	*clicks per hour at position 1* [(a)]	*price at position 1 ($)*
Clep test	0.400	0.20
Learn java	0.200	0.93
College credit for service	0.015	0.10
Transfer credit	0.015	0.11
Master degree online	0.600	5.50
Military education	0.015	0.36
Master in education	0.075	2.55
Online master degree program	0.058	3.04
Education degree online	0.010	2.90

Notes: [(a)] Position 1 clicks was estimated from clicks model. 'clep test' and 'learn java' generate a large number of clicks for a small outlay in cost.

expenditure dropped slightly from $18.41 per day to $16.68 per day during the experimental period. The new expenditure is closer to the targeted spending, although this was not statistically significant.

We did however, discover a surprising change in the *variance* in daily expenditure. In the experimental period, the standard deviation in daily expenditure decreased from $10 to $5 (Figures 13 and 14).

This was an unexpected result, but makes sense because the agent is adjusting each hour for spending under- and over-shoots. For instance, let's say that during hour 1 we plan to spend $500 for the next day.

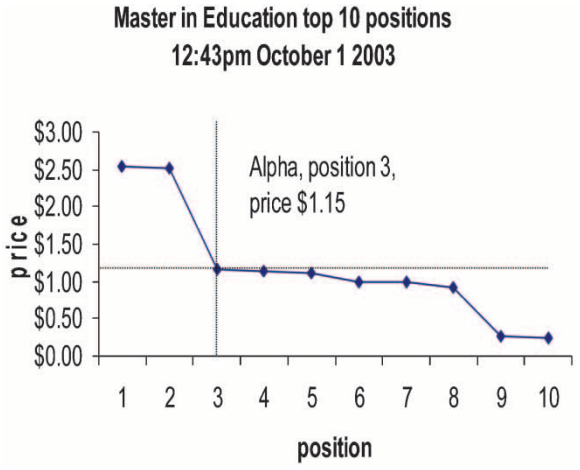

Figure 17. Top 10 positions for the keyword auction for 'Master in Education' at 12:43pm on 1 October 2003. This shows how the agent seems to naturally find and situate itself into gaps, without any 'gap finding rule' needing to be in effect

However, we get a rush of customers and use up \$100 in the first hour. In hour 2, the agent re-computes its allocation plan, and this time only plans to spend \$500/24 – \$100 + \$500 = \$420.83 across the next 24 hours, starting at hour 2. Thus, in general, if expenditure is higher than expected, the agent throttles back on its budget. If expenditure is lower than expected, the agent throttles forward — electing to use the unexpected funds in the best way possible. We call this compensatory behaviour 'feedback stabilization'. The decreased variance is certainly a welcome result. Auctions are unpredictable, and no manager would want to be surprised by a budget blow-out at the end of the quarter.

RELATED WORK

Although there has not been a lot of published research on PPC auctions, a diverse body of work exists on trading agents in other online auction formats. The earliest work in this area was related to trading on the capital markets (Freedman *et al.* 1995, Skabar and Cloete 2001, 2002). A recent focus of research has been the annual Trading Agent Competition (TAC), begun in 2000 in an effort to test, compare and research different trading agent strategies (Wellman *et al.* 2003a, 2003b, 2004). Many fascinating strategies have been developed for this competition, including variants on AdaBoost (Schapire *et al.* 2002), Constraint programming (Aurell *et al.* 2002), and simple rules (Fritschi and Dorer 2002). Kephart *et al.* (2000) have written at length on the concept of PriceBots — agents that search for the cheapest price for a variety of online goods. Etzioni *et al.* (2003) have developed a trading agent to speculate on online airline prices. The system observes patterns of price movement in airline tickets and decides how far ahead to buy a ticket so as to secure the cheapest price. This is another system which has substantial commercial implications.

Our work is similar in orientation to work on eBay auctions. Recently there has been a proliferation of interest in analyzing bidding strategies (Shah *et al.* 2002), developing trading agents (Bapna 2003), and procuring multiple goods through this online auction format (Bapna *et al.* 2000). Much like PPC auctions, a large number of commercial software products are being sold to automate bidding, often using timed bidding strategies called 'sniping'. In addition, the revenue tied up in eBay auctions, like PPC, is staggering. Despite the similarities, the auction mechanism in PPC is vastly different from eBay (and for that matter, TAC and PriceBots). PPC is a second price, continuous auction, in which 'everybody wins', whereas eBay is a first-price auction in which a participant either wins or loses. The theoretical differences are profound — whereas eBay auctions have a Nash equilibrium, sealed bid PPC auctions do not. The theoretical properties of PPC auctions are examined in Kitts and LeBlanc (2004).

CONCLUSION

We have presented an intelligent agent for bidding on PPC keyword search auctions. The agent develops a future look-ahead bidding plan that enables it to hold back cash for more desirable times of the day. The agent neatly melds traditional keyword-level rules with global optimization, so that users can exercise specific control over individual keywords, constrain the optimizer to particular positions and so on. The agent is also designed to handle model degeneracy, automatically repair its models, and generally be as self-sustaining as possible. The agent has been tested on a real PPC auction, where it quadrupled the clicks for the same expenditure.

PPC auctions are ideal marketing vehicles, since as long as the value of a customer is greater than the minimum bid O, marketers can place bids that are guaranteed to make them a profit. However, there is a need to systematically manage PPC auctions so that the maximum possible profit is achieved. We believe that optimization-based approaches have an advantage over keyword-specific and rule-based management methods, especially in managing large numbers of keywords and believe these methods will become increasingly used in the future.

Notes

1. GoToast calls these ROI rules, KeywordMax calls them Optimization rules.
2. Based on the client's data for a typical campaign, profit was calculated to be approximately \$2.96 per click. The estimated profit, based on this number, rose from \$13.52 +/– \$10.71 to \$101.00 +/–

$38.39. However, this is unrealistically high, and the actual profit would have been dependent upon signup rates and registration rate. Therefore we really do not know the profit generated by this experiment, and it would be misleading to believe these figures. As a result we focus on clicks generated versus cost in our analysis of results.

References

eBay Inc. (2003) *Form 10-K, Annual Report Persuant to Section 13 or 15(d) of the Securitites Exchange Act of 1934 for the Fiscal Year Ended December 31, 2003.* File number 000-24821. www.sec.gov

Overture Services Inc. *Form 10-Q, Quarterly Report Persuant to Section 13 or 15(d) of the Securities Exchange Act of 1934 for the Quarterly Period Ended June 30, 2003*, United States Securities and Exchange Commission. File number 000-26365. www.sec.gov

Aurell, E., Boman, M., Carlsson, M., Eriksson, J., Finne, N., Janson, S., Kreuger, P. and Rasmusson, L. (2002) 'A Constraint Programming Agent for Automated Trading', in *Eighth International Conference of the Society for Computational Economics: Computing in Economics and Finance*, Aix-en-Provence, France, June.

Bapna, R. (2003) 'When Snipers Become Predators: Can Mechanism Design Save Online Auctions?', *Communications of the ACM* 46(12) December: 152–8.

Bapna, R., Goes, P. and Gupta, A. (2000) 'A Theoretical and Empirical Investigation of Multi-item On-line Auctions', *Information Technology and Management* 1: 1–23.

Berry, D. and Fristedt, B. (1985) *Bandit Problems: Sequential Allocation of Experiments*, London, UK: Chapman and Hall.

Brandt, F. and Weiß, G. (2001a) 'Antisocial Agents and Vickrey Auctions', *Proceedings of the Eighth International Workshop on Agent Theories, Architectures*, Seattle.

Brandt, F. and Weiß, G. (2001b) 'Vicious Strategies for Vickrey Auctions', *Proceedings of the Fifth International Conference on Autonomous Agents*, Montreal, Canada.

Carroll, R. J. and Ruppert D. (1988) *Transformation and Weighting in Regression*, New York: Chapman and Hall.

Etzioni, O., Knoblock, C., Tuchinda, R. and Yates, A. (2003) 'To Buy or Not to Buy: Mining Airline Fare Data to Minimize Ticket Purchase Price', *Proceedings of the Ninth International Conference on Knowledge Discovery from Databases*, Washington, DC.

Freedman, R., Klein, R. and Lederman, J. (eds) (1995) *Artificial Intelligence in Capital Markets*, Chicago, IL: Probus Publishers.

Fritschi, C. and Dorer, K. (2002) 'Agent-oriented Software Engineering for Successful TAC Participation', in *First International Joint Conference on Autonomous Agents and Multi-agent Systems*, Bologna.

Girosi, F., Jones, M. and Poggio, T. (1993) 'Priors, Stabilizers and Basis Functions: From Regularization to Radial, Tensor and Additive Splines, AI Memo 1430, CBCL Paper 75, http://www.ai.mit.edu/people/girosi/home-page/memos.html

Holland, J. (1975) *Adaptation in Natural and Artificial Systems*, Ann Arbor: University of Michigan Press.

Kaebling, L. (1993) *Learning in Embedded Systems*, Cambridge, MA: MIT Press.

Kaebling, L., Littman, M. and Moore, A. (1996) 'Reinforcement Learning: A Survey', *Journal of Artificial Intelligence Research* 4: 237–85.

Kalton, G. and Kasprzyk, D. (1986) 'The Treatment of Missing Data', *Survey Methodology* 12: 1–16.

Karur, S. and Ramachandran, P. (1995) 'Augmented Thin Plate Spline Approximation in DRM', *Boundary Elements Communications* 6: 55–8, online at: http://wuche.wustl.edu/~karur/papers.html

Kephart, J., Hanson, J. and Greenwald, A. (2000) 'Dynamic Pricing by Software Agents', *Computer Networks*, extended version online at: http://www.research.ibm.com/infoecon/researchpapers.html

Kitts, B. and LeBlanc, B. (2004) 'Non-equilibrium Bidding on Keyword Auctions', Technical report, iProspect, Arlington, MA.

Little, R. and Rubin, D. (1987) *Statistical Analysis with Missing Data*, New York: John Wiley and Sons.

Masters, T. (1995) *Neural, Novel & Hybrid Algorithms for Time Series Prediction*, New York: John Wiley & Sons.

Montgomery, A. and Rossi, P. (1999) 'Estimating Price Elasticities with Theory-based Priors', *Journal of Marketing Research* 36(4): 413–23.

Pasternack, D. (2002) 'Advanced PPC Bidding Strategy', *DidIt White Paper*, 10 December.

Pasternack, D. (2003) 'Rapid Response Predictive Bidding', *DidIt White Paper*, 25 April.

Ryan, T. P. (1997) *Modern Regression Methods*, New York: Wiley.

Sandholm, T. (2000) 'Issues in Computational Vickrey Auctions', *International Journal of Electronic Commerce* 4(3): 107.

Saroop, A. and Bagchi, A. (2002) 'Artificial Neural Networks for Predicting Final Prices in eBay Auctions', *12th Workshop on Information Technology and Systems*, Barcelona, Spain, 14–15 December.

Schafer, J. (1997) *Analysis of Incomplete Multivariate Data*, London: Chapman and Hall.

Schafer, J. (1999) 'Multiple Imputation: A Primer', *Statistical Methods in Medical Research* 8: 3–15.

Schapire, R., Stone, P., McAllester, D., Littman, M. and Csirik, J. (2002) 'Modeling Auction Price Uncertainty using Boosting-based Conditional Density Estimation', *Proceedings of the Nineteenth International Conference on Machine Learning.*

Shah, H., Joshi, N., Sureka, A. and Wurman, P. (2002 forthcoming) 'Mining for Bidding Strategies on eBay', *Lecture Notes on Artificial Intelligence.*

Skabar, A. and Cloete, I. (2001) 'Discovery of Fincancial Trading Rules', *Proceedings of Artificial Intelligence and Applications*, 121–5.

Skabar, A. and Cloete, I. (2002) 'Neural Networks, Financial Trading and the Efficient Markets Hypothesis', *Proceedings of the 25th Australasian Computer Science Conference*, Melbourne, Australia, pp. 241–9.

Specht, D. F. (1991) 'A General Regression Neural Network', *IEEE Transactions on Neural Networks* 2: 568–76.

Stone, P., Littman, M. L., Singh, S. and Kearns, M. (2001) 'ATTac-2000: An Adaptive Autonomous Bidding Agent', *Journal of Artificial Intelligence Research* 15: 189–206.

Thrun, S. (1992) 'The Role of Exploration in Learning Control', in White, D. and Sofge, D. (eds), *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*, New York: Van Nostrand Reinhold.

Wellman, M., Greenwald, A., Stone, P. and Wurman, P. (2003a) 'The 2001 Trading Agent Competition', *Electronic Markets* 13(1): 4–12.

Wellman, M., Cheng, S., Reeves, D. and Lochner, K. (2003b) 'Trading Agents Competing: Performance, Progress and Market Effectiveness', *IJCAI 2003 Workshop on Trading Agent Design and Analysis.*

Wellman, M., Reeves, D., Lochner, K. and Vorobeychik, Y. (2004) 'Price Prediction in a Trading Agent Competition', *Journal of Artificial Intelligence Research* 21: 19–36.

Wurman, P. Wellman, M. and Walsh, W. (2001) 'A Parameterization of the Auction Design Space', *Games and Economic Behavior* 35: 304–38.

Wurman, P., Wellman, M. and Walsh, W. (2002) 'Specifying Rules for Electronic Auctions', *AI Magazine* 23(3): 15–23.