Co-learning Segmentation in Marketplaces

Edward Robinson[1], Peter McBurney[2], and Xin Yao[1]

[1] School of Computer Science, University of Birmingham,
Edgbaston, Birmingham B15 2TT, UK.
e.r.robinson,x.yao@cs.bham.ac.uk
[2] Department of Informatics, King's College London,
Strand London WC2R 2LS, UK.
peter.mcburney@kcl.ac.uk

Abstract. We present the problem of *automatic co-niching* in which potential suppliers of some product or service need to determine which offers to make to the marketplace at the same time as potential buyers need to determine which offers (if any) to purchase. Because both groups typically face incomplete or uncertain information needed for these decisions, participants in repeated market interactions engage in a learning process, making tentative decisions and adjusting these in the light of experiences they gain. Perhaps surprisingly, real markets typically then exhibit a form of parallel clustering: buyers cluster into segments of similar preferences and buyers into segments of similar offers. For computer scientists, the interesting question is whether such co-niching behaviours can be automated. We report on the first simulation experiments showing automated co-niching is possible using reinforcement learning in a multi-attribute product model. The work is of relevance to designers of online marketplaces, of computational resource allocation systems, and of automated software trading agents.

1 Introduction

In a famous 1929 paper in economics, Harold Hotelling showed that competing sellers in a marketplace may end up offering to their customers very similar products to one another [6]. Imagine that potential customers for ice-cream are distributed uniformly along a beach. If there is only one supplier of ice-creams to these customers, the rational location for his or her ice-cream stand is in the middle of the beach, since this minimizes the average distance that customers would need to walk to reach the stand. If, however, there are two competing ice-cream suppliers, the rational location for these two suppliers is right beside each other in the middle of the beach. This is because this location means that each supplier maximizes the number of potential customers for whom his or her stand is the nearest. Any other position for the two seller-stands means that for one seller more than half the potential customers are closer to the other stand; that seller therefore has an incentive to move closer to the middle of the beach. Given a fixed and uniform distribution of potential customers, then, the final position of the two sellers will be side-by-side in the middle of the beach. If the distribution of potential customers is not uniform but has a finite mean, then the final position of the two sellers will be side-by-side at this mean.[3]

[3] assuming that both sellers are rational and both know the distribution of potential customers.

Most suppliers would prefer not to be located immediately beside their direct competitors in the conceptual space of their product category. Indeed, one view of marketing as an activity is that rational marketers always seek to differentiate a product (or service, or a supplier) in the minds of target customers from alternative products (or services, or suppliers) or alternative means of satisfying the need satisfied by the product, and to differentiate sufficiently that a premium can be charged or that would-be competitors are deterred [10]. In addition, because for most product categories customer preferences are complex and diverse — only rarely are they arrayed along a single dimension, as in Hotelling's model — the range of possible product positionings (seller locations) is immense and their rational selection by a seller complex. Not only would a rational seller deciding its product positions consider the distribution of preferences of its potential customers, but also the varied costs and technical requirements of provisioning different offers and positionings, along with the known or likely offers and positionings of competitors.[4] Many of these factors essential for a rational positioning decision are unknown to a seller ahead of launch of the product, particularly for those products which create a new product category (e.g., new technologies). As a consequence, seller positioning is often an incremental learning process, in which one or more offers are made to potential customers, with the reactions of buyers, potential buyers, and competitors then being observed, and then the offers modified, withdrawn or replaced with others. As a result, what we see in many market categories is a process of self-organization of sellers, in which the sellers gradually settle into a situation where each seller offers different products or services to different target segments of customers, in a form of dynamic positioning.

But potential customers too may be ignorant of relevant aspects of the marketplace. Customers need to learn not only what offers are being made by which suppliers, but also their own preferences across these offers. For new product categories, a customer's preferences may not be known even to themselves in advance of purchase and use. For so-called *network goods* — goods (such as fax machines or networking protocols) where the utility gained by one user depends on the utilities of other users — rational potential customers need to know the preferences of other customers in order to determine their own preferences. Thus, customers as well as suppliers may be engaged in incremental co-learning and self-organization, in a form of dynamic segmentation.

Thus, marketplaces typically exhibit a process of dynamic and incremental co-learning (or co-evolution, or co-self-organization) of product positions and customer segments, with suppliers learning from customers and from other suppliers, and also customers learning from suppliers and from other customers. These parallel learning processes inform, and are informed by, each other. This phenomenon has also been seen in the games of the *CAT Market Design Tournament*, where entrants compete to provide exchange services for automated software traders [2, 9]. However, the client-server architecture of the CAT Tournament makes it impossible to know to what extent entrant strategies are controlled by humans or are automated. For computer scientists interested in software trading and online marketplaces, the question arises whether and to what extent these co-learning processes can be automated. This paper reports on

[4] See [5] for a detailed guide to market positioning decisions in just one product category, that of mobile-phone services.

the first work undertaken in this domain, work which has application for the design of computational resource allocation systems as well as to automated marketplaces.

This paper is structured as follows. Section 2 presents a formal model of a computational marketplace in which sellers offer multi-attribute products or services to potential buyers. Potential buyers have different preferences over these attribute-bundles, and so need to decide which to purchase. Likewise, sellers have different costs of provision and need to decide which attribute bundles to offer. Section 3 discusses in detail what we call the *automatic co-niching* problem, and considers reinforcement learning approaches to tackle it. Section 4 then reports on a simulation study to compare strategies for automatically locating market niches, and Section 5 concludes the paper.

2 A model of multi-attribute resource allocation

This section presents the model describing the distributed approach to multi-attribute resource allocation via a set $\mathcal{M}$ of distributed competing double auction marketplaces, which are able to choose the type of resource to be traded within their market, while a set of traders $\mathcal{T}$ trade in the resource markets that most suit their preferences and constraints. While other models and platforms for studying competition between marketplaces exist, e.g., JCAT [2], they only consider single-attribute resource allocation across marketplaces. Thus, the work presented here is motivated by the need for a new model of both trader and marketplace behaviour, which will enable study of the proposed approach, because unlike previous models: *(i)* the resources are multi-attribute in nature, and traders have preferences and constraints over them; and *(ii)* marketplaces have to specifically choose what types of multi-attribute resources can be traded within their market.

2.1 Abstract Computational Resources

Many types of computational resource can be accurately specified in terms of a bundle of *attributes*, because of their often quantifiable nature. In this model we consider *abstract* computational resources, only assuming that a resource comprises a vector $\boldsymbol{\pi}$ of n non-price attributes:

$$\boldsymbol{\pi} = \langle \pi_1, \pi_2, \ldots, \pi_n \rangle \,, \tag{1}$$

where $\pi_i \in [0, 1]$ refers to the *attribute-level* of the ith attribute. Resources can be differentiated by their *type*, which is defined by the levels of each of their attributes. Two resources can be considered identical *iff* all of their attribute-levels are equal, i.e., $\boldsymbol{\pi}^1 \equiv \boldsymbol{\pi}^2 \iff \forall_j, \pi^1_j = \pi^2_j$. Different consumers will have varying minimum resource requirements, which must be satisfied in order that the resource is useful to them. Realistically, these requirements might fall upon a minimum level of storage or random-access memory for large data-oriented tasks, or processing power for time-sensitive tasks. A user can impart these requirements on their trading agent a_i using a vector $\mathbf{r}^{a_i}_{\lfloor}$ of *minimum constraints*:

$$\mathbf{r}^{a_i}_{\lfloor} = \left\langle r^{a_i}_{\lfloor 1}, r^{a_i}_{\lfloor 2}, \ldots, r^{a_i}_{\lfloor n} \right\rangle ,$$

where $r^{a_i}_{\lfloor j}$ is, for example, the minimum level attribute j must meet in order to be useful to a_i. As well as minimum constraints, consumers might *not* require certain attribute to be above specific thresholds, e.g., because their tasks only require a certain amount of memory to run. Likewise, providers may have constrained hardware or capacity, and may only be able to provide certain attribute-levels to consumers; a user's laptop-based resource has different maximum attribute-levels to a node on a high-speed computational cluster, for example. Again, these requirements can be communicated to trading agents via a vector $\mathbf{r}^{a_i}_{\uparrow}$ of *maximum constraints*:

$$\mathbf{r}^{a_i}_{\uparrow} = \left\langle r^{a_i}_{\uparrow_1}, r^{a_i}_{\uparrow_2}, \ldots, r^{a_i}_{\uparrow_n} \right\rangle,$$

where $r^{a_i}_{\uparrow_j}$ is the maximum constraint on attribute j, and $\forall_j, r^{a_i}_{\uparrow_j} \geq r^{a_i}_{\lfloor j}$. As well as expressing preferences over different resources, multi-attribute decision theory states that decision-makers might have preferences over the individual attributes of a resource [7]. For consumers, represented by buying agents, preferences describe the relative importance of each attribute, in terms of value. For providers, represented by selling agents, preferences describe the relative *cost* of providing each of the attributes. It is assumed each trader a_i maintains a vector $\mathbf{w}^{a_i}$ of preferences over the attributes of a resource:

$$\mathbf{w}^{a_i} = \langle w^{a_i}_1, w^{a_i}_2, \ldots, w^{a_i}_n \rangle,$$

where $\forall_j, w^{a_i}_j > 0$ and $\sum_{j=1}^{n} w^{a_i}_j = 1$. If the trader a_i does not have preferences over the different attributes, equal weighting is applied to all attributes: $\mathbf{w}^{a_i} = \left\langle \frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n} \right\rangle$.

2.2 Agent Decision Making Models

Within this system, buyers and sellers need to have a decision-making model that allows them to state their preferences over various multiple-attribute computational resources. By using a multi-attribute utility function, an agent's preferences over the types of resources defined above can be quantified, allowing a decision maker to get a conjoint utility measure for each multi-attribute resource, based upon each of the individual attribute utilities, by combining them according to relative importance.

Trader multi-attribute utility functions Previous agent-based computational resource allocation models, e.g., [1], have proposed that agents make use of the additive multi-attribute utility function introduced by Keeney and Raiffa [7], which, using their notation, is of the form $u(\mathbf{x}) = \sum_{i=1}^{n} k_i u_i(x_i)$, where u and u_i are the utility functions for the entire resource $\mathbf{x} = \langle x_1, x_2, \ldots, x_n \rangle$ and each individual attribute x_i respectively. The utility of each attribute is weighted according to its preferences or importance to the decision maker; the weight of attribute i is represented by k_i. However, additive functions of this type, while combining attribute utilities according to relative importance, fail to consider one important computational resource assumption, viz., that worthless resources, with attributes failing to satisfy minimum constraints, should provide zero utility. It is clear that no matter what the utility of individual attributes, it is not possible for one attribute x_i to determine the entire resource utility. In order that computational

resource consumers' constraints on minimum attribute levels can be realistically modelled, we introduce a richer utility function that enforces the assumptions about buyers' preferences over resources with attributes that fail to meet these constraints. Formally, a buyer b_i's valuation of a resource $\boldsymbol{\pi}$ is determined according to the following multi-attribute valuation function $\upsilon_{b_i}(\boldsymbol{\pi})$:

$$\upsilon_{b_i}(\boldsymbol{\pi}) = \lambda_{b_i} \left[\sum_{j=1}^{n} w_j^{b_i} \; u_{b_i}(\pi_j) \right] \times \prod_{j=1}^{n} H(\pi_j) \tag{2}$$

Equation 2 has two main parts. The first part of the equation is an additive multi-attribute utility function of the type introduced by Keeney and Raiffa, which determines the contribution of each of the attributes of $\boldsymbol{\pi}$, weighted by their importance according to $w_j^{b_i}$. Because it is assumed that all attribute-levels lie on the range $[0, 1]$, and that $\sum_{w \in \mathbf{w}^{b_i}} w = 1$, the conjoint utility of a resource $\boldsymbol{\pi}$ is naturally scaled between zero and one. It is assumed the utility of a resource to a buyer monotonically increases with the level of its attributes, implying that the weighted attribute utilities of the most desirable resource sums to one. It is also assumed that a buyer would be *indifferent* between an amount of money equal to its budget constraint, λ_{b_i}, and the most desirable resource. Thus, by scaling the utility of a resource by λ_{b_i}, a buyer can state its valuation in terms of money. The second part of Equation 2 ensures that a resource $\boldsymbol{\pi}$'s utility collapses to zero if any attributes fail to satisfy minimum constraints, regardless of the other attribute utilities. This is achieved by checking every attribute satisfies its minimum constraint using a Heaviside step function:

$$H_{b_i}(\pi_j) = \begin{cases} 1 & \text{if } \pi_j \geq r_{\downarrow j}^{b_i} \\ 0 & \text{otherwise,} \end{cases}$$

where $r_{\downarrow j}^{b_i}$ is buyer b_i's minimum constraint for the jth attribute. The utility contribution of each individual attribute is calculated according to b_i's attribute utility function $u_{b_i}(\pi_j)$.

$$u_{b_i}(\pi_j) = \begin{cases} \pi_j & \text{if } r_{\downarrow j}^{b_i} \leq \pi_j \leq r_{\uparrow j}^{b_i} \\ r_{\uparrow j}^{b_i} & \text{if } \pi_j > r_{\uparrow j}^{b_i} \\ 0 & \text{if } \pi_j < r_{\downarrow j}^{b_i} \end{cases}, \tag{3}$$

where $r_{\uparrow j}^{b_i}$ refers to b_i's maximum constraint. $u_{b_i}(\pi_j)$ ensures that if an attribute has a level in excess of a b_i's maximum constraint, it contributes no more utility than if $\pi_j = r_{\uparrow j}^{b_i}$. Sellers, being resource providers rather than consumers, are modelled slightly differently to buyers. Each resource type $\boldsymbol{\pi}$ involves a *cost of production*, defined by a seller's cost function:

$$c_{s_j}(\boldsymbol{\pi}) = \lambda_{s_j} \sum_{i=1}^{n} w_i^{s_j} \; u_{s_j}(\pi_i), \tag{4}$$

where $u_{s_j}(\pi_i)$ is the cost contribution of each of the attributes of $\boldsymbol{\pi}$, weighted by their relative costs according to $w_i^{s_j}$; given two attributes x and y, if $w_x^{s_j} > w_y^{s_j}$ then it costs

more to produce a given increase in attribute x than it does in attribute y. The attribute cost function $u_{s_j}(\pi_i)$ is defined as follows:

$$u_{s_j}(\pi_i) = \begin{cases} \infty & \text{if } \pi_i > r^{s_j}_{\restriction_i} \\ \pi_i & \text{otherwise} \end{cases} \tag{5}$$

Thus, a seller is unable to provide a resource with attributes that exceed its maximum production constraint. In all other cases, the cost of production increases linearly with the attribute level.

Agent payoffs Within a double auction environment, the profit or payoff a buyer or seller gains from a transaction is dependent on the type of resource $\boldsymbol{\pi}$ exchanged, the amount of money τ exchanged (transaction price), and any associated market-exchange costs determined by the market-exchange, which will be communicated to each trader as a vector of costs $\mathbf{c}$. When a transaction takes place, the buyer b_i's payoff P_{b_i} is:

$$P_{b_i}(\boldsymbol{\pi}, \tau, \mathbf{c}) = \upsilon_{b_i}(\boldsymbol{\pi}) - \tau - \sum_{c \in \mathbf{c}} c, \tag{6}$$

while for a seller s_j:

$$P_{s_j}(\boldsymbol{\pi}, \tau, \mathbf{c}) = \tau - c_{s_j}(\boldsymbol{\pi}) - \sum_{c \in \mathbf{c}} c \tag{7}$$

In both cases, because agents are assumed to be able to express all their preferences via money, the size of the payoff is equivalent to an equally sized increase in utility.

Market-exchanges, as with trading agents, are considered utility-maximisers within this model. A market-exchange's utility is measured according to the revenue generated from charging fees to traders. Each market-exchange m_k maintains an *exchange member* set $\mathcal{E}_{m_k} \subset \mathcal{T}$, containing the traders that have joined its market at the beginning of that trading day. During each trading day, m_k also stores all of the transactions θ that it executes, maintaining a transaction set Θ_{m_k}, containing all the transactions that took place that day. An exchange's daily profit P_{m_k} is determined both by the amount of traders that entered the market, and the transactions that the exchange executed:

$$\begin{aligned} P_{m_k}(\mathcal{E}_{m_k}, \Theta_{m_k}) = \\ |\mathcal{E}_{m_k}| \cdot \zeta^{m_k}_{reg} + \sum_{\theta \in \Theta_{m_k}} 2 \cdot \theta_\tau \cdot \zeta^{m_k}_{tra} + [\theta_{bid} - \theta_{ask}] \cdot \zeta^{m_k}_{com}, \end{aligned} \tag{8}$$

where $\zeta^{m_k}_{reg} \in \mathbb{R}_{\geq 0}$, $\zeta^{m_k}_{tra} \in [0, 1]$ and $\zeta^{m_k}_{com} \in [0, 1]$ refer to m_k's *registration fee*, *transaction price* fee and *spread commission* fee levels respectively. Registration fee revenue depends on the number of traders that joined m_k's market that day. Both the buyer and seller pay a transaction price fee to m_k, based upon the transaction price θ_τ. Finally, the spread commission fee is based on the difference between the buyer's bid θ_{bid}, and the seller's ask θ_{ask}.

2.3 Agent Mechanics

Market-exchange agents operating within this resource allocation approach use two main mechanisms: *(i)* a double action mechanism for allocating resources between buyers and sellers; and *(ii)* a mechanism for deciding what type of resource will be traded within their market each trading day. The method by which a market-exchange decides on the attribute-levels of the type of resource to be traded within its market is determined by its *attribute-level selection* (ALS) strategy. In Section 3 we will discuss the *automatic market co-niching* problem—a challenging reinforcement learning problem that market-exchanges face in this type of system, as well as ALS strategies for tackling it.

This work does not concern itself with the design and analysis of policies or rules pertaining to the running of a double auction *per se*. As such, several previously well-defined double auction policies are used by the market-exchanges within this work. These include even ($k = 0.5$) k-pricing policies, continuous market matching and clearing, two-sided quoting and beat-the-quote shout-accepting policies. Finally, market-exchanges make use of fixed charging policies (discussed further in Section 3.2).

Trading agent mechanics Trading-agents are typically composed of two main parts [3]: *(i)* a *trading strategy* that dictates at what price the buyer or seller shouts offers into the market; and *(ii)* a *market-selection strategy* that dictates at which market to enter each trading day. We now outline the strategies used, and how they are adapted or extended for use in our model of multi-attribute resource allocation. In terms of the trading strategy, we assume that all traders use the Zero-Intelligence Plus (ZIP) trading strategy [4]. The two main reasons for this choice are: *(i)* the ZIP strategy has been extensively analysed in double auction settings [4, 3] and found capable of achieving efficient allocations [4]; and *(ii)* the ZIP trading strategy is computationally simple, and thus scales well for use in large-scale experiments.

The ZIP algorithm uses a deterministic mechanism that decides which direction (if at all) the agent should adjust its current shout price, while a further part of the algorithm comprises a machine learning technique that decides by what amount to adjust the trader's current shout price. Typically, other applications of ZIP within the literature fail to incorporate the notion of fees that market-exchanges may charge traders. Therefore, we extend a part of the ZIP algorithm to incorporate charges and fees, meaning traders won't trade at a loss. ZIP traders maintain a *limit price*, which for a seller specifies the minimum price they will sell a resource for, or for a buyer specifies the maximum price they will buy a resource for. Limit prices are equivalent to resource valuations, i.e., $\upsilon(\boldsymbol{\pi})$. However, if traders pay registration fees or other transaction based fees, and the transaction price is particularly close the traders' limit prices, then they may make a loss. To prevent this from happening we incorporate the relevant market-exchange fees into traders' limit price calculations. For a buyer b_i, its adjusted limit price $\widehat{\upsilon}_{b_i}(\boldsymbol{\pi})$ is calculated as follows:

$$\widehat{\upsilon}_{b_i}(\boldsymbol{\pi}) = \left[\upsilon_{b_i}(\boldsymbol{\pi}) - \zeta_{reg}^{m_k}\right] \times \left[1 + \zeta_{tra}^{m_k}\right]^{-1}, \tag{9}$$

while for a seller s_j, its adjusted limit price $\widehat{\upsilon}_{s_j}(\boldsymbol{\pi})$ is calculated as follows:

$$\widehat{\upsilon}_{s_j} = \left[\upsilon_{s_j}(\boldsymbol{\pi}) + \zeta_{reg}^{m_k}\right] \times \left[1 - \zeta_{tra}^{m_k}\right]^{-1} \tag{10}$$

In both Equation 9 and 10 $\zeta_{reg}^{m_k}$ and $\zeta_{tra}^{m_k}$ refer to the registration and transaction price fees that market-exchange m_k charges (described in Section 3.2).

For the second aspect of a trading agent—the market-selection strategy—we use a consumer theoretic approach. Modern consumer theory [8] supposes that resource-constrained consumers, being rational and time-constrained (and processing-power-constrained and memory-constrained), only consider a subset of all options available [12]. Some options are immediately rejected without consideration, because they have below-threshold values on essential attributes (so-called *inept* options). Only the contents of the subset of options left, termed the *consideration set* [12], are then carefully deliberated over, before ultimately one option is chosen.

Within our model, each trader's *market-selection strategy* forms a daily consideration set $\mathcal{C}$ of market-exchanges. Market-exchanges are excluded from a trader's consideration set if the resource type offered in the market is considered inept by the trader. Buyers consider resources inept if one of the attribute-levels fails to meet its minimum constraint, while sellers consider resources inept if one of the attribute-levels is beyond their production ability, i.e., maximum constraints. Thus, for a buyer b_i:

$$\mathcal{C}_{b_i} = \{m_k \in \mathcal{M} : (\forall \pi_j \in \boldsymbol{\pi})(\pi_j \geq r_{\lfloor j}^{b_i})\},$$

where $r_{\lfloor j}^{b_i}$ is b_i's minimum constraint for the jth attribute of the resource $\boldsymbol{\pi}$ specified by market exchange m_k. And, for a seller s_j , its consideration set $\mathcal{C}_{s_j}$:

$$\mathcal{C}_{s_j} = \{m_k \in \mathcal{M} : (\forall \pi_j \in \boldsymbol{\pi})(\pi_j \leq r_{\lfloor j}^{s_j})\}$$

Once a consideration set is formed, a more careful evaluation can be made. Because each exchange will have potentially different charges and fees, and each market populated with differing trader types and supply and demand schedules, each trader faces an exploration/exploitation learning problem—trying to learn, over time, the best market-exchange to join each day. In line with the literature [2], we treat this problem as an *n-armed bandit problem*. Each trader a_i maintains a vector of *reward* values:

$$\mathbf{R}^{a_i} = \left\langle \mathrm{R}_{m_1}^{a_i}, \mathrm{R}_{m_2}^{a_i}, \ldots, \mathrm{R}_{m_{|\mathcal{M}|}}^{a_i} \right\rangle$$

Thus, traders maintain a reward value $\mathrm{R}_{m_k}^{a_i}$ for each market-exchange $m_k \in \mathcal{M}$; initially at time $t = 0$, $\forall_{m_k},\ \mathrm{R}_{m_k}^{a_i}(t) = 0$. If a trader a_i joins m_k on day t, it updates its reward value associated with m_k according to:

$$\mathrm{R}_{m_k}^{a_i}(t+1) = \mathrm{R}_{m_k}^{a_i}(t) + \delta_{a_i} \cdot [P_{a_i}^{t} - \mathrm{R}_{m_k}^{a_i}(t)], \tag{11}$$

where $P_{a_i}^{t}$ refers to a_i's profit for trading day t, and δ_{a_i} to a discounting factor that a_i uses to ensure that more recent profits contribute further towards $\mathrm{R}_{m_k}^{a_i}$, i.e., $\mathrm{R}_{m_k}^{a_i}$ becomes an *exponential moving average*. The ϵ-*greedy* strategy selects the market-exchange with the highest reward with probability ϵ, while a random market-exchange is chosen with probability $1 - \epsilon$ times. Thus, ϵ represents the probability of exploitation (joining the historically best market-exchange), while $1 - \epsilon$ represents the probability of exploration. In case of ties, a_i chooses randomly between market-exchanges.

2.4 The Trading Process

Finally in this section we give the reader a general outline of the trading process, from the view of both traders and market-exchanges. Within our market-based system, we assume the following stages occur within each *trading day*: *(i) attribute-level selection stage*—at the beginning of the trading day each exchange defines the type of resource to be traded in its market by broadcasting the resource's attribute-levels. *(ii) daily market-selection stage*—next, traders decide which of the market-exchanges to join; traders may only join one exchange per trading day. *(iii) trading and trader learning stage*—the trading day is split into a number of *trading rounds* (opportunities to shout offers into the market). *(iv) venue learning stage*—at the end of the trading day traders and market-exchanges calculate their daily profit. This is used as a signal to the decision mechanisms that dictate behaviour on the next trading day.

3 The automatic co-niching problem

The previous section formally introduced a model of multi-attribute resource allocation via competing marketplaces. Within such an approach, resources are allocated via distributed markets, using double-auction mechanisms run by market-exchange agents. A significant new question is: *how should market-exchange agents best select which types of resources should be traded in their markets?* This section considers for the first time the *automatic market co-niching problem*, where market-exchanges must *autonomously* select the types of resources to be traded within their markets, in the presence of other competing and *coadapting* market-exchanges doing the same. Using two reinforcement learning approaches, several algorithms, which we call *attribute-level selection* (ALS) strategies, are considered for tackling the problem.

The *automatic co-niching problem* can be summarised as follows. At the beginning of each trading day, an exchange must define the type of multi-attribute resource that can be traded within its single market by setting and broadcasting a vector of attribute-levels forming a resource type $\boldsymbol{\pi} = \langle \pi_1, \pi_2, \ldots, \pi_n \rangle$. A trader a_i prefers markets for resources that best align with its preferences ($\mathbf{w}^{a_i}$) and maximum and minimum constraints ($\mathbf{r}^{a_i}_{\uparrow}$) and ($\mathbf{r}^{a_i}_{\downarrow}$). Further, a reasonable assumption is that while traders' preferences and constraints can be unique, cohorts of traders exist within *market segments*. Different market segments prefer to trade different resources, for example a segment of traders working on behalf of data-centres or backup services may be more interested in trading high-storage computational resources. A natural consequence of competition between traders is that they will migrate to markets that most satisfy their segment.

In order to attract traders and generate trades, exchanges therefore need to identify resource types that best satisfy market segments. The process of discovering these segments is called *market niching*, and the product or service that satisfies a market segment is called a *market niche*. Thus, the automatic market co-niching problem is one of finding market niches via searching the *attribute-level space* for vectors of attributes that form resource which satisfies a market segment. However, as discussed in Section 1, automating the search for market niches is particularly challenging. Firstly, it is a *co-niching* problem because multiple competing exchanges are attempting to do the same, which can cause competition over niches or otherwise change the learning

problem. Further, it is unlikely that a single algorithm, in the form of an ALS strategy, would perform best over all possible environments, because the environment is complex, adaptive, and coevolving. However, progress can be made on this problem by identifying what impact different environmental factors have on strategies for market niching, and specifically what approaches work well and why.

3.1 Attribute-level Selection Strategies

Market-exchanges' ALS strategies are required to systematically search the attribute-level space, looking for niches where market-exchanges can maximise their profits (generated from fees and charges). Given the environment is dynamic and coevolving—because of other market-exchanges' decisions, and trading agents' learning—the typical revenue generated from traders *changes over time*, and exchanges must constantly explore the attribute-level space to identify the most lucrative types of resource markets.

Over a number of trading days, defined as an *evaluation period*, each ALS strategy evaluates a single resource type (a real-valued vector of attributes) by providing a market for that resource. Evaluation periods of sufficient length help to dampen oscillations in daily market profits caused by the dynamic nature of the environment. Once the evaluation period is finished, the reward in terms of the mean daily market profit over the period is recorded, and a new resource type chosen by the ALS strategy through the selection of new attribute-levels. We consider in this paper two approaches for ALS strategy design.

Market niching as a multi-armed bandit problem The first approach we consider is to treat the automatic niching problem as a *multi-armed bandit* (MAB) problem. A MAB models a world where an agent chooses from a finite set of possible *actions*, and executing each action results in a *reward* for the agent. In the simplest MAB problem, the distributions associated with each lever do not change over time [11], though some variations allow the reward distribution of the pulled lever to change once pulled. However, what sets the *automatic niching problem* apart from these situations is that the reward distributions of *unchosen actions* can change over time too, making the bandit problem *restless* [17]. For example, an action with poor rewards over some time horizon may have excellent rewards during some future time horizon.

To deal with the continuous attribute-level space we discretise it and assume each resource attribute π_j can take $n = 5$ distinct levels: $\forall_{\pi_j \in \boldsymbol{\pi}}\ \pi_j \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$. A non-zero minimum level is chosen because in reality, most if not all computational resources need at least some level of each attribute to be desirable. Given q attributes, there are $n^q = 25$ possible two-attribute resource types and each market-exchange m_k's ALS strategy maintains an *action set* $\mathbf{\Pi} = \{\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_{q^n}\}$ of all possible actions. Each ALS strategy maintains with this set of actions an associated *reward set*, $\mathbf{Q}^{m_k} = \left\langle Q^{m_k}_{\boldsymbol{\pi}_1}, Q^{m_k}_{\boldsymbol{\pi}_1}, \ldots, Q^{m_k}_{\boldsymbol{\pi}_{q^n}} \right\rangle$, which is updated after every evaluation period. We explore the application of this approach by implementing several different bandit-based ALS strategies.

The first of these is the ϵ-*greedy* strategy, which explores the environment and chooses a random action from $\mathbf{\Pi}$ with probability ϵ, while selecting the best action (the action with the highest corresponding reward value in $\mathbf{Q}^{m_k}$, where m_k is the market-exchange using the strategy) from $\mathbf{\Pi}$ with probability $1-\epsilon$. Over time, ϵ is fixed, meaning the amount of exploration a market-exchange does is fixed (this makes it known as a *semi-uniform strategy*); for all simulations in this paper $\epsilon = 0.1$, which is a commonly chosen value [16].

The second bandit-based ALS strategy we consider is the ϵ-*decreasing* strategy, which works in an identical way to the ϵ-*greedy* strategy, with the exception that ϵ *decreases* over time according to $\epsilon_t = \min(\delta/t, 1)$, where t is the trading day and $\delta \in [0, +\infty)$ is a schedule set by the user; in all simulations in this paper $\delta = 0.15$ was experimentally found to be a good choice.

The third strategy considered is the *Softmax* strategy. Semi-uniform strategies, when exploring, choose actions with historically bad rewards as often as any other. This can be detrimental when the worst actions are *very* bad. Market-exchanges using *Softmax* avoid these very bad actions by choosing all actions with probability *proportional* to the associated rewards in $\mathbf{Q}^{m_k}$. Each action $\boldsymbol{\pi}_i$ is selected with probability:

$$\psi_{\boldsymbol{\pi}_i} = \frac{e^{Q^{m_k}_{\boldsymbol{\pi}_i}/T}}{\sum_{j=1}^{|\mathbf{Q}|} e^{Q^{m_k}_{\boldsymbol{\pi}_j}/T}}, \tag{12}$$

where $Q^{m_k}_{\boldsymbol{\pi}_i}$ refers to the m_k's historical reward for action $\boldsymbol{\pi}_i$. The temperature T shapes the distribution; when a high temperature is chosen, action choice is approximately equal-probable, while lower temperatures widen the probability gap between choosing different actions. For all simulations in this paper, $T = 0.3$ was chosen experimentally.

Finally, the fourth strategy considered is the *Rank-based* strategy. This strategy is inspired by the *rank selection* often used to maintain diversity in genetic algorithms. Like *Softmax*, the probability of choosing an action is proportional to its historical rewards, however, the probability of choosing it is independent of the *quantitative* value of the historical reward, only its performance rank, relative to the others. Thus, in the case of action $\boldsymbol{\pi}_i$, the probability of it being chosen is:

$$\psi_i = \texttt{rank}(\boldsymbol{\pi}_i)^{\varsigma} / \sum_{j=1}^{n} \texttt{rank}(\boldsymbol{\pi}_j)^{\varsigma}, \tag{13}$$

where ς, the selection pressure, again controls the tradeoff between exploration and exploitation. The function $\texttt{rank}(\boldsymbol{\pi}_i)$ outputs the rank of action $\boldsymbol{\pi}_i$ based upon its historical reward $Q^{m_k}_{\boldsymbol{\pi}_i}$; the action with the best historical reward is ranked $|\mathbf{Q}|$, while the action with the lowest ranked 1. For all the bandit strategies discussed, the rewards $Q \in \mathbf{Q}$ are decayed over time because the problem of finding market niches is clearly *non-stationary*. Specifically, a market-exchange m_k can update the reward $\mathrm{Q}^{\mathrm{m_k}}{}_{\boldsymbol{\pi}_i}$ for action $\boldsymbol{\pi}_i$ in the next time-step $t+1$ as follows:

$$Q^{m_k}_{\boldsymbol{\pi}_i}(t+1) = Q^{m_k}_{\boldsymbol{\pi}_i}(t) + \delta\left[r^t_{\boldsymbol{\pi}_i} - Q^{m_k}_{\boldsymbol{\pi}_i}(t)\right], \tag{14}$$

where δ is a discount and $r^t_{\boldsymbol{\pi}_i}$ is the *instantaneous reward* returned by the π_i in time-step t; in this model, that equates to the profit the market-exchange made on trading day t.

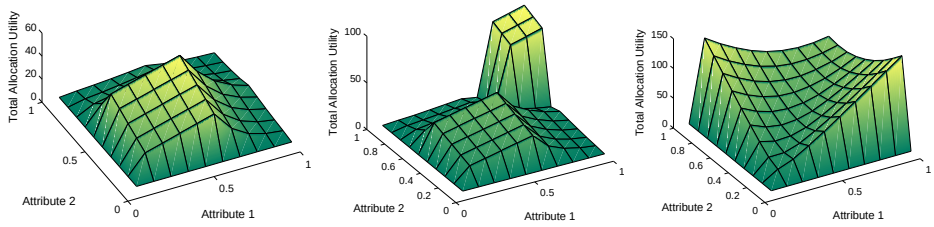
Market Niching as an optimization problem Reducing the number of possible resources an ALS strategy can choose from, through discretising the attribute-level space, can be useful for effective exploration. However, if there is a relationship between points in the attribute-level space, and the rewards those points provide, bandit strategies cannot leverage this information, as they do not consider the relationship between actions in the attribute-level space. Evolutionary optimization algorithms work on the principle that improving solutions are often found close by, so algorithms tend to search in and around neighbouring points; this is often appropriate if the fitness function being optimised is continuous. However, in some environments these algorithms may become stuck in *local optima*, hindering their progress. ALS strategies using an evolutionary optimization approach can be deployed if the set of possible resource types is defined as a set of real-valued vectors: $\mathbf{\Pi} = \{\boldsymbol{\pi} : \forall_{\pi_j \in \boldsymbol{\pi}},\ \pi_j \in \mathbb{R}_{\geq 0}\}$. Using this definition, evolutionary algorithms can then evolve arbitrary resource types, rather than being confined to choose from a small set (as bandit-based ALS strategies are). The profit that a market-exchange receives from specifying a resource type $\boldsymbol{\pi}_i$ becomes the *fitness* assigned to $\boldsymbol{\pi}_i$.

For this initial investigation, we consider two basic evolutionary optimization ALS strategies. The first is the *1+1 ES* ALS Strategy, which is a simple *evolutionary strategy* that maintains a population size of two, consisting of the current best individual (the parent), and a candidate next solution (the offspring). Each individual represents a resource type $\boldsymbol{\pi}$ in the form of a vector of two attribute-levels, where $\forall_{\pi_j \in \boldsymbol{\pi}},\ \pi_j \in [0.2, 1.0]$. When this attribute-level selection strategy is used, a new offspring individual $\boldsymbol{\pi}_o$ is generated each evaluation period using a mutated copy of the parent $\boldsymbol{\pi}_p$. Mutation is carried out through perturbing each attribute-level $\pi_j \in \boldsymbol{\pi}_p$ by a value drawn from the Gaussian distribution $\mathrm{N}(\pi_j, \sigma)$, with standard deviation σ. The offspring is used as the resource type for the exchange's market during the next evaluation period, and if its fitness is larger than the parent's, it becomes the new parent. Through some initial exploratory simulations we settled on a value of $\sigma = 0.12$ for the *1+1 ES* ALS strategies used in this work.

The second strategy is called the *EA* ALS strategy. Unlike the the *1+1 ES*, it is assumed the *EA* algorithm maintains a population size of greater than two at all times. For this work, we consider *EA* ALS strategies that maintain populations of ten individuals. Selection among the individuals is carried out by using a process called *tournament selection*, where (in this case three) resource types (individuals) are evaluated in the environment and the two with higher fitnesses are combined to replace the weaker.

3.2 Representative Environmental Contexts

In general, the performance of market mechanisms can be sensitive to a number of environmental factors, and thus market mechanisms can be seen to be robust (or obversely, brittle) to different environments, as Robinson et al. [13] showed, using their



(a) Constrained Single Niche (b) Constraint-induced Niches (c) Preference-induced Niches

Fig. 1: Three different *trader contexts*. (a) The *Constrained Single Niche* context only has one population of traders interested in the same niche. Buyers have maximum constraints on both attributes, which creates a market niche at the point for resource $\boldsymbol{\pi} = \langle 0.6, 0.6 \rangle$. (b) The *Constraint-induced Niches* trader context comprises two trader sub-populations and two market niches. One population prefers to trade resources with high-level attributes—the most desirable types being at the market niche for resource $\boldsymbol{\pi} = \langle 1.0, 1.0 \rangle$. The other population, due to maximum constraints, most desires to trade resources $\boldsymbol{\pi} = \langle 0.6, 0.6 \rangle$. (c) The *Preference-induced Niches* trader context also comprised two trader sub-populations and two market-niches. Unlike the *Constraint-induced* trader context the niches within this context are formed from the preferences traders have over resource attributes. One subpopulation prefers to trade resources $\boldsymbol{\pi} = \langle 1.0, 0.2 \rangle$, while the other subpopulation prefers to trade resources $\boldsymbol{\pi} = \langle 0.2, 1.0 \rangle$.

methodology for measuring the generalization properties of market mechanisms. The principal approach of the methodology is to identify the main building blocks of the environment—the notions that *define* the environment—and generate a set of *representative environments*, from these. This is particularly useful in this work's environment, where evaluating the mechanisms described here, in all possible environmental conditions, is impractical.

Within this paper, the same methodology is applied so that the performance and impact of various attribute-level selection strategies can be empirically analysed. The model of resource allocation considered assumes an environment defined in three ways: *(i)* by the general makeup of the trading population, particularly in terms of their preferences and constraints over resources (*trader context*); *(ii)* by the charging schemes used by the market-exchanges, which affects the behaviour of traders within the trading population (*charging context*); and *(iii)* both the presence of, and the strategies in use by, competing market-exchanges (*competitor context*). Each of these individual *contexts* affect and change the overall *environmental context*. The last of the contexts, the *competitor context* is defined by the presence or absence of other competing agents within the environment. The other two contexts are now discussed in more detail.

Trader contexts In terms of the trader context, in this paper we present results using three different contexts, examples of which are shown in Figure 1. Importantly, the trader contexts we use for these simulations cover situations where there are less market niches than there are market-exchanges (Figure 1a), as well as when there are multiple niches (Figures 1b–1c). Each trader context contains a number of market niches and

market-exchanges' ALS strategies must explore the attribute-level space to find these. The landscapes in Figure 1 show, under ideal conditions, the maximum utility that could be generated if markets for the resource types described by the x and y axes existed. Market-exchanges can be expected to generate revenue proportional to the height of the peaks. However, these landscapes are only ideal representations. In reality the height of these peaks can change throughout simulations as competing market-exchanges, as well as the trader population, learn. Further, the charging scheme used by the market-exchanges can have a significant impact on their ability to locate these niches and the amount of revenue each generates. More details on how these trader contexts are formed can be found in Chapter 5 of [15].

Charging contexts The profit a market-exchange receives by offering a market for a certain resource type is influenced not only by that resources location in the attribute-level landscape, but also by the charging scheme used to generate that profit. As described in Section 2.3 it is assumed for these introductory investigations that market-exchange make use of three types of charges. We consider three charging contexts in this paper where each of the charges is used in isolation, so that we may see the impact it has on the exchanges ability to locate market niches.

The first charging context considered is the *Registration Fee* context. Market-exchanges using this context charge each trader that joins their exchange each day a fixed amount; for all simulations within this paper the registration fee $\zeta_{reg}^{m_k} = 0.01$. The second charging context considered is the *Transaction Price Fee* context. Market-exchanges using this context only charge traders when they successfully transact in the market. Specifically, each trader is charged a percentage of the transaction price of the trade; for all simulations within this paper, the percentage $\zeta_{tra}^{m_k} = 0.01$. The third charging context considered is the *Bid/Ask Spread Commission* context. Market-exchanges using this context only charge traders who successfully transact a portion of the *spread*—the difference between their shout and the transaction price. Because this charging context only taxes traders on profit, they will never make a loss in a market using this charging context. For all simulations within this paper, the percentage charged $\zeta_{com}^{m_k} = 0.01$.

4 Agent-based simulation study

In this section we carry out a significant computational study of the market-based system described so far in this paper, and specifically on the applicability and performance of *attribute-level selection strategies* with respect to tackling the *automatic co-niching problem* in bilateral simulations of competing market-exchanges. Firstly, we briefly describe the general setup used throughout. Every simulation last for 5000 trading days, and the mean values from 50 repetitions of each simulation variant are reported. The trading population used in each simulation comprises 300 trader and is composed of an equal number of buyers and sellers; depending on the trader context being used the constraints and preferences of the traders may differ. For each simulation repetition, along with a new *random seed*, all traders' budget constraints were randomly generated according to the normal Distribution $\mathcal{N}(6, 0.7)$ creating new supply and demand schedules each time.

	ϵ-dec	ϵ-gre	EA	1+1 ES	Rank-based	Softmax
ϵ-dec		**2.477** 0.70 (0.01) 0.206 (0.83)	**3.585** **1.329** **0.302**	**3.157** **1.120** **0.259**	**3.527** **1.096** **0.309**	1.878 **1.310** **0.303**
ϵ-gre	1.720 **0.916** (0.01) 0.205 (0.82)		**3.449** **1.361** **0.255**	**3.111** **1.029** **0.220**	**3.256** **1.125** **0.283**	1.529 **1.372** **0.273**
EA	0.292 0.232 0.056	0.252 0.201 0.049		0.449 0.523 (0.07) 0.060 (0.02)	0.642 0.421 0.099	0.053 **0.625** 0.101
1+1 ES	1.027 0.461 0.090	0.831 0.543 0.098	**2.358** 0.753 (0.07) 0.107 (0.04)		**1.932** (0.2) 0.764 0.089 (0.90)	0.645 **0.962** **0.128**
Rank-based	0.444 0.496 0.151	0.435 0.496 0.141	**1.239** **1.006** **0.146**	1.365 (0.2) 0.668 (0.36) 0.094 (0.90)		**1.326** **1.089** **0.124**
Softmax	2.586 (0.02) 0.260 0.036	**2.722** 0.197 0.025	**3.634** 0.438 0.079 (0.07)	**3.442** 0.287 0.042	0.637 0.345 0.084	

Table 1: Mean *simulation profit* for market-exchanges involved in bilateral simulation using various ALS strategies in environments containing the *Constrained Single Niche* trader context. Each profit value belongs to a market-exchange using the ALS strategy on that *row*, in competition with an exchange using the ALS strategy in that *column*. Each cell has three profit values, representing simulations where one of the charging contexts was in use: (top) *Transaction Price Fee* charging context; (middle) *Registration Fee* context; and (bottom) *Bid/Ask Commission* context. Thus, the value in the absolute top-right of the table (1.878) represents *ϵ-decreasing*'s mean profit in simulations against *Softmax*, when the *Transaction Price Fee* charging context was in use. Emboldened values indicate the result is greater than the competitor's and the samples are statistically distinct. All *p-values* less than 0.005 are omitted, otherwise they are shown to the right of profit values.

Due to the complex nature of interactions between these economic agents it is unwise to assume that data samples will be normally distributed. To overcome this assumption rigorous statistical analysis is carried out. To test for normality, all data samples are subjected to the *Lilliefors Test*, a goodness of fit test for the Normal distribution. If samples are found to be non-normally distributed then they are compared using the non-parametric *Wilcoxon Signed-rank Test*, otherwise a *paired sample T-Test* is used.

4.1 Experimental Results

The empirical analysis is carried out in two main parts. In the first part we consider competition between two market-exchanges over a single market niche. Many simulation variations are carried out, using different combinations of charging and trader context, in order to ascertain the ability of various ALS strategies for finding and holding onto a market niche. In the second part of the study we consider competition between two market-exchanges in environments where there are multiple market niches. Again

many combinations of environmental contexts are considered, however the emphasis is on the *co-niching* ability of the market-exchanges, and what impact contexts have on the allocative efficiency of the entire system.

Competition over single market niches In this set of simulations we consider the *Constrained Single Niche* trader context, as shown in Figure 1a. Specifically, for this trader context we run simulations of all environmental contexts formed from all combinations of the charging contexts and competitor pairings; this results in 90 environmental contexts and some $90 \times 50 = 4500$ simulations (each simulation variant is repeated 50 times).

Performance of ALS strategies is measured quantitatively by looking at the profit that each market-exchange makes over the life of simulation, when using each of the ALS strategies to choose the resource types to be traded within its market. Self-play simulations, where both exchanges use the same strategy are not considered, because results in expectation would be identical in these cases.

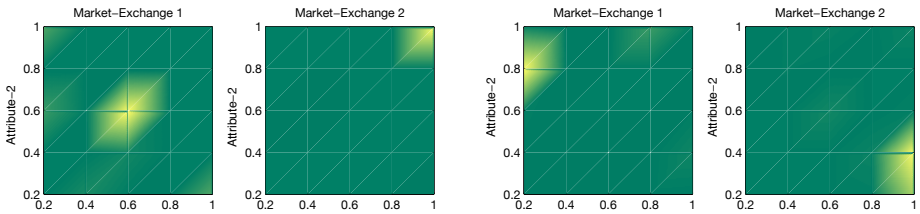
All results for this set of experiments are shown in Table 1. In general the reader will note that the emboldened profit values towards the top of the table indicate that the two semi-uniform bandit strategies located there, ϵ-*greedy* and ϵ-*decreasing*, performed the strongest out of the ALS strategies in almost all of the environmental contexts. However, it is clear that there are some strong performances by other ALS strategies in *certain circumstances*. For example, when the *Transaction Fee* charging context was in place (top cell row) the *Softmax* ALS strategy (bottom row of table) performed statistically better than ϵ-*greedy* and others. This suggests that *Softmax* is particularly sensitive to the charging context in use, but that its proportional exploration may be better than ϵ-greedy's under ideal conditions. Finally, it is clear that the evolutionary optimization approaches do not perform as well as the bandit algorithms. This is a particularly interesting result, and closer examination of data traces from individual simulations reveals that in single niche environments, the competing market-exchange is able to 'flatten' the profit landscape by attracting all of the traders to its exchange. In such a situation the *1+1 ES* and *EA* ALS strategies are unable to navigate the attribute-level landscape because all points return no fitness (due to no traders joining their exchange). While they can attract traders back, often they are unable to due to being in poor parts of the attribute-level landscape, and relying on moving to neighbouring points to make progress.

Market co-niching in multi-niche environments In this set of simulations, rather than looking directly at the profitability of the ALS strategies, we look at the niching behaviour of the strategies, and the overall allocative efficiency of system. Of particular interest is whether two market-exchanges, in competition with each other, are able to locate and satisfy multiple market niches, leading to efficient and stable allocations. For this set of experiments we consider the two multi-niche trader contexts: *Constraint-induced Niches* and *Preference-induced Niches*, as shown in Figure 1b–1c. Again, all bilateral combinations of ALS strategies were run within the two trader contexts, and each variant repeated for different charging contexts. Further, self-play simulations were

considered as in a multi-niche environment how a strategy interacts—either competitively or cooperatively—is important.

Allocative efficiency results are shown in Table 2 for all simulation variations. The allocative efficiency metric measures directly the amount of utility (social welfare) generated in the system each trading day and compares it to the theoretical maximum, which is calculated using a bespoke optimization algorithm (see Chapter 4 [15]). In reality it is not possible for the efficiency to be continually close to 1.0, because of the complex nature of the system.

We find from our simulations that in general no charging context leads to the most efficient allocations across all possible environmental contexts. In general we note that while the *Transaction Price Fee* charging context is generally better for environments with the *Constraint-induced* trader context, the *Bid/Ask Commission* charging context is preferable when the *Preference-induced* trader context is present. The *Registration Fee* charging context is never preferable is one wishes to maximiser efficiency. This is most likely because market-exchanges have little incentive to locate niches precisely as all points within the general location of a niche will result in about the same number of traders joining the exchange (and thus similar revenues). Finally, we again note that the presence of either of the two semi-uniform bandit strategies tends to result in the higher allocative efficiencies seen.



(a) (b)

Fig. 2: *Heat Maps* showing typical niching behaviour of several attribute-level strategies. The maps aggregate the decisions of the ALS strategies over many thousands of trading days. Lighter areas indicate that the point in the attribute-level space was chosen more frequently. (a) Typical performance of two market-exchanges each using a *Softmax* ALS in the *Transaction Price Fee* charging and *Constraint-induced* trader contexts. Two niches exist: $\boldsymbol{\pi} = \langle 0.6, 0.6 \rangle$ and $\boldsymbol{\pi} = \langle 1.0, 1.0 \rangle$ (see Figure 1b). (b) Typical performance of two ϵ-*greedy* ALS strategies in the *Preference-induced* trader context along with the *Bid/Ask Spread Commission* charging context. The two niches are located at $\boldsymbol{\pi} = \langle 0.2, 1.0 \rangle$ and $\boldsymbol{\pi} = \langle 1.0, 0.2 \rangle$ (see Figure 1c).

For the final analysis of the multi-niche environments we provide the reader with visualisations that allow one to get an understanding of the typical niching behaviour for various environment contexts. In Figure 2 we provide visualisations of two typical simulations. The first simulation, Figure 2a, is taken from a simulation where the *Constraint-induced* trader context was used, and where the *Transaction Price Fee* charging context was in place. Two market-exchanges (each using the *Softmax* ALS strategy) are com-

	Allocative Efficiency		
Strategy	Bid/Ask Spread charging context	Registration Fee charging context	Transaction Fee charging context
ϵ-dec	0.60 ± 0.08 ***0.57 ± 0.11***	*0.57 ± 0.11* 0.42 ± 0.05	***0.72 ± 0.07*** 0.49 ± 0.03
ϵ-gre	*0.63 ± 0.09* **0.56 ± 0.09**	0.57 ± 0.12 *0.42 ± 0.06*	**0.70 ± 0.08** *0.51 ± 0.04*
EA	0.40 ± 0.13 **0.40 ± 0.09**	0.36 ± 0.11 0.32 ± 0.05	**0.49 ± 0.12** 0.40 ± 0.05
1+1 ES	0.44 ± 0.13 0.47 ± 0.09	0.43 ± 0.11 0.35 ± 0.05	**0.59 ± 0.15** **0.48 ± 0.05**
Rank	0.55 ± 0.11 **0.50 ± 0.04**	0.50 ± 0.10 0.37 ± 0.06	**0.59 ± 0.09** 0.43 ± 0.05
Soft	0.37 ± 0.15 0.38 ± 0.08	0.31 ± 0.12 0.31 ± 0.03	**0.69 ± 0.11** **0.48 ± 0.06**
All	0.48 ± 0.16 **0.48 ± 0.12**	0.45 ± 0.16 0.36 ± 0.07	**0.63 ± 0.14** 0.46 ± 0.06

Table 2: For each strategy, each data point provides the measure of system-wide mean daily allocative efficiency, over *all* simulations involving each strategy and its competitors (including against itself). Thus, it captures the impact that the presence of that strategy typically has on resource allocations within the system. Results are separated into the three different charging contexts, thus each value is the mean from a sample of data points with size: 6 competitor variations $\times$ 50 simulation repetitions. A value of 1.0 would indicate a 100% efficient allocation for every day of every reported simulation—a very unlikely outcome given the complex and dynamic nature of the system. Results are further separated according to the trader context in use. The top value in each cell refers to the *Constraint-induced* trader context, while the bottom value in each cell refers to the *Preference-induced* trader context. For each ALS strategy and trader context, emboldened values indicate highest reported efficiency across the three charging contexts. Emphasised values indicate the highest reported efficiency for *any* strategy involved within that single charging and trader context.

peting over two market niches, and as one can see from the the visualisation, they both settle on separate niches, leading to a very efficient allocative efficiency ($\sim 78\%$) and a profitable outcome for both. The *Transaction Price Fee* charging context allows the ALSs to accurate locate the market niches.

In the second simulation, Figure 2b, we show a situation where the *Preference-induced* trader context is in place, and the *Bid/Ask Commission* charging context is used by the exchanges, which are both using *ϵ-greedy* ALS strategy. In this case the market-exchanges get quite close to the niches, but don't precisely satisfy them (allocative efficiency was $\sim 64\%$). Interestingly, this may be due to market-exchanges generating more revenue from less efficient markets with this charging context; thus by staying slightly away from the optimal market niche point, a wider spread and a larger commission can be maintained.

5 Conclusion

This paper has described a phenomenon which is familiar to people in Marketing: sellers decide their product positions incrementally, and in parallel with potential buyers deciding which offers to accept. Both sellers and buyers make their choices in the absence of the full information needed to make these decisions rationally, and so, typically, each engages in a dynamic learning process. Buyers typically cluster into segments of similar types or behaviours, while sellers typically position themselves to make offers to some segments and not others. Thus, we witness a process of co-learning or co-self organization, with both sellers and buyers clustering in response to one another and influencing one another. The long-term situation, typically, is that certain suppliers target certain customer segments (and not others), and certain customers are loyal to certain suppliers (and not others): the ecology of the marketplace is thus a collection of smaller niches, with suppliers competing directly with only *some* of the other suppliers in that market, and customers only ever considering purchases of *some* of the offers made by suppliers. For designers of electronic marketplaces, the question arises whether this co-niching behaviour can be automated.

The simulation study reported here has shown that such automation is possible, using reinforcement learning of attribute-level selection (ALS) strategies. The study was undertaken by running more than 10,000 bilateral market-based simulations comprising hundreds of traders with differing preferences and constraints. In general we found that all ALS strategies considered are sensitive to at least some environmental factors, and thus none can be seen to generalize across all contexts, but in most environmental contexts, at least one of the strategies performs very well. In particular we have shown that it is possible for market-exchanges to autonomously locate market niches within these types of environments. We have also identified that strategies that rely on quantitative reward values, e.g., Softmax, can be brittle and sensitive to parametric settings in these environments, while strategies that only rely on qualitative comparisons of rewards from actions, e.g., ϵ-greedy and ϵ-decreasing, are more robust. Further, we note that evolutionary approaches to attribute-level selection were not as successful as bandit approaches. While we believe this is caused by the manipulation of the fitness landscape by competitors, a more detailed investigation into this phenomena, as well as seeking to improve the robustness properties of parametrically-sensitive strategies will be future work.

The multi-attribute decision model presented in Section 2 of this paper greatly simplifies the actual market positioning and consumer selection decisions of real marketplaces [8]. With the problem of automated co-niching now identified, further work will be required to extend these results to more realistic models of marketplaces. Such an agenda forms part of a computational science of the dynamics of marketplaces, and becomes increasingly important as our economic life moves online.

Acknowledgments

This research was partly funded by the UK EPSRC through the *Market Based Control of Complex Computational Systems* project (GR/T10671/01 and GR/T10657/01). This paper arose from an invitation to the second author to present an invited talk at the *2011*

International Workshop on Adaptive and Learning Agents (ALA 2011), held in Taipei, Taiwan. We thank the organizers of ALA 2011 for this invitation, and the participants at that workshop for their comments and questions. We also acknowledge insightful discussions on these topics with Martin Chapman, Peter Lewis, Tim Miller, Jinzhong Niu, Simon Parsons and Steve Phelps. The technical work reported here forms part of the first author's PhD thesis [15] and some of these results were also presented at ICEC2011 in Liverpool, UK in August 2011 [14].

References

1. M. Bichler and J. Kalagnanam. Configurable offers and winner determination in multi-attribute auctions. *European Journal of Operational Research*, 160(2):380–394, 2005.
2. K. Cai, E. Gerding, P. McBurney, J. Niu, S. Parsons, and S. Phelps. Overview of CAT: A Market Design Competition (version 2.0). Technical Report ULCS-09-005, Department of Computer Science, University of Liverpool, Liverpool, UK, 2009. Version 2.0.
3. K. Cai, J. Niu, and S. Parsons. On the economic effects of competition between double auction markets. In *10th Workshop on Agent-Mediated Electronic Commerce (AMEC 2008).*, Estoril, Portugal, 2008.
4. D. Cliff and J. Bruten. Zero is Not Enough: On The Lower Limit of Agent Intelligence For Continuous Double Auction Markets. Technical Report HPL 97-141, HP Laboratories Bristol, 1997.
5. P. Hemming, D. Shuker, and P. McBurney. *Innovative PCS Marketing: How to Build and Execute a Winning Marketing Plan*. Redwing Consulting Ltd, Dallas, TX, USA, 1996.
6. H. Hotelling. Stability in competition. *The Economic Journal*, XXXIX:41–57, 1929.
7. R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley & Sons, NY, USA, 1976.
8. G. L. Lilien, P. Kotler, and K. S. Moorthy. *Marketing Models*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1992.
9. J. Niu, K. Cai, S. Parsons, P. McBurney, and E. Gerding. What the 2007 TAC Market Design Game tells us about effective auction mechanisms. *Autonomous Agents and Multi-Agent Systems*, 21(2):172–203, 2010.
10. M. E. Porter. *Competitive Advantage: Creating and Sustaining Superior Performance*. Free Press, New York, NY, USA, 1985.
11. H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.
12. J. H. Roberts and G. L. Lilien. Explanatory and predictive models of consumer behavior. In J. Eliashberg and G. L. Lilien, editors, *Handbooks in Operations Research and Management Science:Volume 5: Marketing*, pages 27–82. North-Holland, 1993.
13. E. Robinson, P. McBurney, and X. Yao. How specialised are specialists? generalisation properties of entries from the 2008 and 2009 TAC market design competitions. In E. D. et al., editor, *Agent-Mediated Electronic Commerce: Designing Trading Strategies and Mechanisms*, volume 59 of *LNBIP*, pages 178–194. Springer, 2010.
14. E. Robinson, P. McBurney, and X. Yao. Market niching in multi-attribute computational resource allocation systems. In W. Ketter, K. R. Lang, and K. J. Lee, editors, *Proc. 13th International Conference on Electronic Commerce (ICEC 2011)*, Liverpool, UK, 2011.
15. E. R. Robinson. *Resource Allocation via Competing Marketplaces*. PhD. available: http://bit.ly/kumsfc, University of Birmingham, UK, Birmingham, UK, 2011.
16. R. Sutton and A. Barto. *Reinforcement learning: An introduction*. The MIT press, 1998.
17. P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, pages 287–298, 1988.