# Governance and Matching*

Tomas B. Klos†

**Abstract**

This paper is concerned with the organization of transactions of goods and services between consecutive stages of activity. An economic theory of organization, transaction cost economics (TCE)—which proposes that organizational form (e.g. market and hierarchy) is adjusted to the attributes of transactions—is extended with the idea that the governance of transactions should be analyzed within the wider network of the firms they connect, and that agents' behavior is guided by adaptive learning rather than by optimization.

An agent-based computer simulation model is developed and experimented with, to study the patterns of governance that emerge from interactions between agents making and breaking relations. In each of a sequence of timesteps, a matching algorithm assigns buyers to suppliers or to themselves, implementing their choices for market and hierarchy, respectively. From each timestep to the next, the agents are allowed to adapt their preferences for each other—that determine the outcome of the matching—to their experiences. Patterns of economic organization are thus 'grown' as the outcome of processes of interaction between boundedly rational agents adaptively searching for 'good' organization.

**Keywords:** governance, trust, matching, artificial adaptive agents

**JEL Classification:** C63, C78, D83, L22

*For helpful comments and discussions, I am grateful to Maryse Brand, René Jorna, Bart Nooteboom, Han La Poutré, Leigh Tesfatsion, Rob Vossen, Michel Wedel and seminar participants at the Research Institute and Graduate School SOM and at the Center for Mathematics and Computer Science (CWI Amsterdam).

†Faculty of Management and Organization, University of Groningen. P.O. Box 800, 9700 AV Groningen, The Netherlands. E-mail: t.b.klos@bdk.rug.nl, WWW: http://www.bdk.rug.nl/medewerkers/t.b.klos

# 1 Introduction

This paper is concerned with the organization of transactions of goods or services between stages of activity. Two consecutive stages might be brought together within a single firm—using hierarchy to organize transactions between the stages; or the different stages could be distributed across separate, specialized firms—using the market to organize transactions between them. Other organizational forms 'between market and hierarchy' could also be used. According to transaction cost economics (Coase 1937, Williamson 1985), the decision between these alternatives is made by 'aligning' organizational form with the attributes of the transaction to be organized.

In the current paper, it is attempted to deal with Coase's (1998, p. 73) recent observation that "[w]e cannot confine our analysis to what happens within a single firm", but that "[w]hat we are dealing with is a complex interrelated structure" (Coase 1995, p. 245). Furthermore, although TCE builds its main 'discrete alignment hypothesis' on the assumption that economic agents are boundedly rational and potentially opportunistic, we submit that it is precisely their bounded rationality that may *prevent* economic agents from performing this alignment successfully, especially in the context of the complex interrelated structures that Coase (1995) suggests we are dealing with. Therefore, instead of searching for optimal mechanisms of governance that firms *should* (but may never) use, we use computer simulations to model the *process* by which agents adaptively search for *satisficing*—rather than optimal—organizational forms, to generate hypotheses about which forms economic agents (come to) use.

The next section discusses governance in more detail. Networks of firms are viewed as *complex systems* containing *adaptive agents* (Holland 1992, Holland and Miller 1991), as discussed in Section 3. Section 4 introduces 'matching', the tool used to build Coase's (1995) complex interrelated structures from individual firms' preferences for different organizational forms. Section 5 develops the computer simulation model in which these complex interrelated structures are build in each of a sequence of timesteps, while firms may adapt their preferences from each timestep to the next. Results from experimentation with the simulation model are presented and discussed in Section 6. Section 7 concludes.

# 2 Governance

The unit of analysis in TCE, as its name suggests, is the transaction. "A transaction occurs when a good or service is transferred across a technologically separable interface. One stage of activity terminates and another begins" (Williamson 1981*a*, p. 552). Rather than focus on individual stages of activity—viewing the firm as a production function to be optimized—TCE

focuses on transactions *between* stages of activity and views the firm as one of the organizational forms that may be used to organize such transactions. The core of the argument is that the firm and the market are *alternative* forms for organizing transactions, that transactions carry costs (transaction costs) and that the various organizational forms have differential abibilities to economize on these costs, so that for some transactions, it is economic to organize them within a firm's hierarchy rather than on the market.

The interchangeability of market and hierarchy in this respect was recognized for the first time by Coase (1937), who received the 1991 Nobel Prize in Economics for his discovery of the significance of transaction costs in elucidating *The Nature of the Firm*; Olson (1965)[1] and Williamson (1975) followed up on Coase's insights, distinguishing market and hierarchy as alternatives. Later on, Williamson (1979, p. 234) additionally acknowledged some "intermediate modes of organization", "in which bilateral dependency conditions are supported by a variety of specialized governance features (hostages, arbitration, take-or-pay procurement clauses, tied sales, reciprocity, regulation, etc.)" (Williamson 1991, p. 269).

Solving governance problems in particular circumstances requires, according to TCE, that organizational form ('governance') is aligned with the attributes of the transaction to be organized, in a discriminating—mainly transaction cost economizing—way; in general, the trade-off should be considered between costs of transaction, organization and production. As for the attributes, a transaction occurs with a certain *frequency*, is surrounded by a certain degree of *uncertainty*, and, most importantly, is supported by investments in assets with a certain degree of *specificity*, i.e. the extent to which those assets can not be redeployed outside the transaction and make sustaining the transaction a necessary condition for obtaining returns on investments in them. Because of the first of TCE's two behavioral assumptions—that agents are *boundedly rational*—contracts are necessarily incomplete, so that before the end of the period during which the transaction needs to be sustained, unforeseen contingencies may arise to which the parties will have to adapt. However, when separate, autonomous firms are involved, then because of TCE's second behavioral assumption—that agents are *potentially opportunistic*—this adaptation can not be assumed to be cooperative (i.e. in their mutual interest), but will rather result in costly haggling over the distribution of the unforeseen gains or losses (Williamson 1981*a*). When organizing a transaction between such separate autonomous firms, therefore, the potential loss of returns on investments in specific assets, as well as the probability of that loss, increase with the specificity of the assets. With increasing specificity, the costs of safeguard-

[1]Olson (1965, p. 12) writes about "economic organizations that are mainly means through which individuals attempt to obtain the same things they obtain through their activities on the market".

ing against the expected loss eventually become so high that the transaction should be removed from the market and organized within the firm, where adaptation is more likely to be cooperative and the costs lower.

# 3 Complex adaptive systems

## 3.1 Complex systems ...

Recently, the founding father of transaction cost reasoning, Coase (1995, p. 245), noted that

> "[t]he analysis cannot be confined to what happens within a single firm. The costs of coordination within a firm and the level of transaction costs that it faces are affected by its ability to purchase inputs from other firms, and their ability to supply these inputs depends in part on their costs of coordination and the level of transaction costs that they face which are similarly affected by what these are in still other firms. What we are dealing with is a complex interrelated structure."

Holland (1992) and Holland and Miller (1991) suggest to study economic systems as 'complex adaptive systems', where a complex adaptive system (CAS) "is a complex system containing adaptive agents, networked so that the environment of each adaptive agent includes other agents in the system" (Holland and Miller 1991, p. 365). The CAS approach thus appears to be ideally suited to deal with Coase's (1995) observation, which is what is attempted in the current paper.

## 3.2 ... and adaptive agents

It is granted that TCE assumes bounded rationality, albeit for the sole purpose of rendering problematic the combination of asset specificity and opportunism; all three are needed as conditions for the existence of firms, i.e. conditions under which the market looses (some or even all of) its advantage because of increasing transaction costs.[2] However, after assuming bounded rationality (for this purpose), TCE goes on to hypothesize *alignment* of transactions with governance structures, while it is precisely their *bounded* rationality that may *prevent* economic agents from successfully performing this alignment, especially in the context of the 'complex interrelated structures' that Coase (1995) suggests we are dealing with.

[2]These conditions, by the way, although necessary, are not necessarily sufficient. The theory says that under these circumstances firms may exist, but it does not explain how they come into existence: it specifies the conditions under which firms have a comparative advantage over markets, but not what is required for this advantage to be translated into the actual emergence of firms (cf. Axtell 1999), which is addressed in the current paper—it can not be done within TCE's conceptual framework.

In this paper, therefore, a different approach than the application of the mathematical logic of economic optimization (or 'alignment') will be taken to generate propositions about how economic activity is organized. Individual, boundedly rational economic agents are simulated in a computational model, along with the decentralized trades they initiate between each other. As in Vriend's (1995, p. 205) model, then, "market interactions depend in a crucial way on local knowledge of the identity of some potential trading partners". The agents themselves decide whether they want to make or buy. Moreover, the option to 'buy' really just consists of a number of alternatives to buy *from*. A market has to be 'made', before it can ever used as a governance form (Vriend 1995, Vriend 1996, Weisbuch *et al.* 1998). Rather than rely on standard, anonymous random matching devices, these decisions are also explicitely incorporated in the model. Agents are assumed to have differential preferences for different potential trading partners (Weisbuch *et al.* 1998).

Economic organization is studied *from the bottom up* (cf. Epstein and Axtell 1996); the resulting distribution of economic activity across different organizational forms *emerges* from processes of interaction between these agents, as they adapt future decisions to past experiences. The system may or may not settle down and if it does, the resulting equilibrium may or may not be transaction cost economic; in any case, "[i]t is the process of becoming rather than the never-reached end points that we must study if we are to gain insight" (Holland 1992, p. 19).

### 3.3 Agent-based Computational Economics (ACE)

"[T]he specialization to economics of the basic Complex Adaptive Systems (CAS) paradigm"[3] described above, goes under the name Agent-based Computational Economics (ACE). This approach is used more and more often to study problems in economics, such as in the repeated prisoner's dilemma (Klos 1999, Miller 1996, Stanley *et al.* 1994), social dilemmas (Glance and Huberman 1994) and on final-goods markets (Albin and Foley 1992, Vriend 1995), stock markets (Arthur *et al.* 1997), industrial markets (Péli and Nooteboom 1997), whole-sale markets (Kirman and Vriend 1998, Weisbuch *et al.* 1998), labor markets (Tesfatsion 1999), spatial political models (Kollman *et al.* 1992, Miller and Stadler 1998), etc. As shown in the current application, the ACE-approach is also very well suited for studying economic organization.

The essence of this approach is that economic phenomena are studied as they emerge from actual (simulated) interactions between individual, boundedly rational, adaptive agents. They are not deduced from abstract models

[3]Quoted from the ACE website, maintained by Leigh Tesfatsion at:
`http://www.econ.iastate.edu/tesfatsi/ace.htm`

employing representative agents, auctioneers or anonymous, random matching, etc. Rather, whether an interaction takes place between any two given agents is left for them to decide. What the agents subsequently do in that interaction is their own—possibly sub-optimal—decision, that they make on the basis of their locally available, incomplete information and as a result of their own (cognitively-limited) processing of that information. Appropriate forms of reasoning are induction and abduction, rather than deduction as used in optimization models that are solved for 'never-reached end points'.

# 4 Matching

The crucial insight underlying our model of firms attempting to solve problems of organization, is that their choices between market or hierarchy can also be seen as the result of a process in which buyers are assigned to suppliers or to themselves, respectively. Such a process, in turn, can be generated by executing a so-called matching algorithm. The simulation model presented in Section 5, therefore, uses such a matching algorithm;[4] the current section describes the algorithm in some detail.

A matching algorithm produces a set of matches (a matching) on the basis of individual agents' preference rankings over other agents. Besides a preference ranking, each agent maintains a 'minimum tolerance level' that determines which other agents are *acceptable*, namely those agents that are somehow 'better' than the agent's minimum tolerance level; agents will not (want to) be matched to other agents they deem unacceptable. Finally, each agent has a *maximum* number of matches it can be involved in at any one time (a quotum).

The algorithm used is Tesfatsion's (1996) deferred choice and refusal (DCR) algorithm, which extends[5] Gale and Shapley's (1962) deferred acceptance algorithm.[6] The DCR algorithm is used with some qualifications. First of all, only *disjoint* sets of buyers and suppliers are allowed, so that there are no agents that can be buyer as well as supplier. So, although buyers may be their own supplier, they can not supply to other buyers. Furthermore, we allow different agents to have different quota—i.e. different maximum numbers of matches allowed at any moment in time—because different buyers and suppliers are likely to want different numbers of partners.

[4]See (Roth and Sotomayor 1990) for an excellent introduction to and overview of matching theory.

[5]To be precise, the DCR algorithm allows both sides of the market to be coincident, overlapping or disjoint, and it also allows arbitrarily specified offer and acceptance quota.

[6]These algorithms produce *stable* matchings, which are matchings that have no blocking (pairs of) agents, i.e. (pairs of) agents who can (bi- or) unilaterally improve upon their actual situation under the matching by—rather than to their actual match—being matched to (each other or) themselves. The DCR algorithm was used because it provides a way of assigning agents to each other, not because it produces stable matchings; in the current application, stability is just a side-effect.

Finally, and most importantly, unlike the DCR algorithm, we *do* allow buyers to be matched to themselves, in which case they are their own supplier. *Each buyer includes itself as one of the alternatives in its preference ranking, and suppliers not ranking higher than the buyer are unacceptable.* This effectively endogenizes the buyer's preferences for different organizational forms; a buyer prefers to remain single (and 'make') rather than 'buy' from an unacceptable supplier. The argument is that buyers on industrial markets don't necessarily need a supplier to make a profit; they can choose to *make* rather than *buy* what they need. On final goods markets, the agents on both sides of the market are qualitatively different from one another: consumers are *individual* people but firms are *groups* of individuals; people can not do certain things that organizations can do. On industrial markets, the agents on both sides of the market are firms, so that a buyer-firm may perform the same functions as a supplier-firm—albeit less efficiently because the buyer does not specialize in performing those functions—and thereby economize on the costs of coordinating the transaction with the supplier-firm; determining whether the buyer-firm should or should not perform a function itself is at the heart of transaction cost economic reasoning.[7]

**The algorithm** Buyers may have one or more suppliers and suppliers may have one or more buyers; each buyer $b$ has an *offer quotum*, $o_b$ ($\geq 1$) and each supplier $s$ has an *acceptance quotum*, $a_s$ ($\geq 1$). Before the matching, all buyers and suppliers establish a strict preference ranking over all their alternatives. The algorithm then proceeds in a finite number of steps.

1. In the first step, each buyer sends a maximum of $o_b$ requests to its most preferred, acceptable suppliers.[8] Because the buyers typically have different preference rankings, the various suppliers will receive different numbers of requests.

2. The suppliers first reject all requests received from unacceptable buyers.[9] Then, each supplier 'provisionally accepts' a maximum of $a_s$ requests from its most preferred acceptable buyers and rejects the rest (if any).

[7]Recently, theories have been developed from the 'competence perspective' (e.g. Nooteboom 1992, Péli and Nooteboom 1997), that stress other arguments for firms to set up relations with other firms. The extensions of transaction cost economics proposed in the current paper also go in that direction.

[8]The algorithm structurally favors the agents that send the requests; buyers seem more plausible than suppliers in that respect.

[9]For the moment, we assume that all buyers are acceptable to the suppliers; suppliers do not, like the buyers, have any alternative, so they will rather supply to any buyer than remain single. It might be investigated, however, whether for a supplier it is worthwhile to also use a tolerance level for protection against being exploited by buyers.

3. Each buyer that was rejected in any step fills its quotum $o_b$ in the next step by sending requests to ($o_b$ minus the number of outstanding, provisionally accepted, requests) next-most-preferred, acceptable suppliers that it has not yet sent a request to.

4. Each supplier again rejects requests received from unacceptable buyers and provisionally accepts the requests from a maximum of $a_s$ most preferred, acceptable buyers from among newly received and previously provisionally accepted requests and rejects the rest. As long as one or more buyers have been rejected, the algorithms goes back to step 3.

The algorithm stops if no buyer sends a request that is rejected. All provisionally accepted requests are then definitely accepted. An example-application of this matching algorithm is presented in Appendix A.1.

# 5 The simulation model

Firms sell a differentiated product on a final-goods market; the simulation model, however, really only captures the industrial market on which these firms are buyers, possibly interacting with suppliers. Governance pertains to the transaction between the production and the sales of the product. Each firm always sells the product himself, and chooses to either produce the product himself, or let a supplier produce it for him, in which case the firm is a buyer. These choices are generated by the DCR matching algorithm: each buyer is either matched to a supplier or to himself, expressing his choice between market- and hierarchichal governance, respectively. The simulation-dynamic refers to the fact that the algorithm is applied in each of a sequence of discrete timesteps. *The outcome of the matching in each timestep is determined by the agents' preference rankings over acceptable alternatives, while the agents may* change *their preference ranking from each timestep to the next.*

After the matching in each timestep, suppliers that are matched to a buyer produce for their buyer(s), while buyers that are 'self-matched' (not matched to a supplier) produce for themselves. Assets that suppliers invest in for the production for a certain buyer, are specific to that buyer to the extent that the buyer's product is differentiated; the remainder of the assets is 'general purpose'. Suppliers enjoy scale-economies in accumulated general purpose assets used in the production for multiple buyers. Furthermore, as their relation lasts longer, a supplier becomes more efficient at using specific assets in the production for a particular buyer. After the production, all buyers sell their products on the final-goods market. The events in this latter part of each timestep—i.e., *after* the matching—may lead the agents to *adapt* their preference rankings, used by the DCR algorithm in the

next timestep. The way preferences are established is described in the next section (5.1). Section 5.2 discusses the implementation of the simulation.

## 5.1 Preferences

The preferences used in the matching process are based on so-called 'scores' that each agent $x$ assigns to all the agents $y$ it can possibly be matched to: $\text{score}_{xy}$ expresses the profit that $x$ *expects* to make as a result of coordinating a transaction with $y$ (in a buyer's case, $y$ may be equal to $x$). It is a function of (1) the profit $x$ can *potentially* make as a result of coordinating the transaction with $y$ and (2) $x$'s *trust* in $y$, which is interpreted as $x$'s assessment of the probability that $y$ will let $x$ *realize* that profit potential—i.e. the probability that $y$ will not behave opportunistically. In order to be able to allow agents to attach different weights to profitability versus trust, however, simple multiplication of the two is turned into a Cobb-Douglas functional form:

$$\text{score}_{xy} = \text{profitability}_{xy}^{\alpha_x} \cdot \text{trust}_{xy}^{1-\alpha_x},$$

where $\text{score}_{xy}$ is the score $x$ assigns to $y$, $\text{profitability}_{xy}$ is the profit $x$ *may* make 'through' $y$, $\text{trust}_{xy}$ is $x$'s trust in $y$ and $\alpha_x \in [0, 1]$ is the importance $x$ attaches to $\text{profitability}_{xy}$ relative to $\text{trust}_{xy}$, i.e. the 'profit-elasticity' of the scores that $x$ assigns. It is the value of $\alpha_x$ that $x$ may adapt from each timestep to the next. The next two sections (5.1.1 and 5.1.2, respectively) describe how profitability and trust are determined.

### 5.1.1 Profitability

A buyer's potential to generate profits for a supplier is a function of the buyer's position on the final market—where he is a seller—as expressed in the degree of *product differentiation* on the market. A supplier's potential to generate profits for a buyer is determined by the supplier's *efficiency* in producing for the buyer.

**Product differentiation** The model allows for varying degrees of product differentiation, i.e. price/cost ratios: not all the firms in an industry sell the same, homogeneous product, or at least, consumers *perceive* different firms' products as being imperfect substitutes. Consumers have idiosyncratic tastes and firms' products have different characteristics, which means that, relative to its competitors' products, each firm's product variant is more or less unique. Because "consumers (...) are prepared to pay more for variants that are better suited to their own tastes" (Anderson *et al.* 1992, p. 1), firms that sell those particular variants have some degree of 'market power': they can raise the price for which they sell their product, without losing at least some of their customers to competitors, as long as the extent

to which their product is better suited to those customers' tastes—relative to their competitors' products—more than offsets the price-increase. This degree of market power will be expressed in a buyer-specific variable $d_b \in [0, 1]$ that determines the profit the buyer will make when selling his products. We will be experimenting with different values for $d_b$ to see how they affect the choices that buyers make.

**Efficiency** As set out above, a buyer's choice of organizational form pertains to the transaction between the production of a product on the one hand, and the sales of that product on the other hand. A buyer will either be self-matched and produce the product himself, or be matched to a supplier who produces it for him. A supplier, on the other hand, may be matched to multiple buyers for which she produces a particular product.[10]

Whoever does it, producing a product requires assets to be invested in—*1 unit of assets is required to produce 1 product, but increasing efficiency may decrease this amount.* Since "asset specificity is never valued by itself but only because demand is thereby increased in design or performance respects" (Williamson 1981*a*, p. 558), we will assume a relation between the differentiation of a buyer's product, and the specificity of the assets invested in to produce that product. The rationale is that, if a buyer $i$'s product is differentiated ($d_i > 0$), then, relative to consumers' tastes, $i$'s product is *different* from his competitors' products. Assets invested in to produce $i$'s product can then not easily be switched to the production of those competitors' (different) products. In other words, those assets are then *specific* to the production of $i$'s product. On the other hand, if products are not differentiated, then they are all the same, and assets invested in to produce the product for one buyer can easily be switched to producing products for other buyers. The simplest way to model this relation, is to assume that asset specificity is *equal* to product differentiation, i.e. the proportion of the asset required to produce a product for a buyer that is specific to that buyer, is equal to the extent to which that buyer's product is differentiated.

If a buyer produces for himself, it makes no sense to distinguish between buyer-specific and non-specific assets.[11] A buyer calculates his own score (his minimum tolerance level) using efficiency = 0, trust = 1 and $\alpha = 1$. If a supplier produces for one or more buyers, however, then the assets she

[10] A more general version of the model would allow for the possibility of multiple components per product and for multiple sources per component. A buyer may then be matched to the same or to different suppliers for the production of the various components; a single supplier may attain economies of scope in the production of different components for the buyer (see Williamson 1981*b*, note 18, p. 1547), whereas multiple suppliers may gain (external) economies of *cognitive* scope in their production for the buyer (cf. Nooteboom 1992, Péli and Nooteboom 1997).

[11] Remember that overlap between both sides of the market is not allowed, which takes away the possibility for buyers to replicate the market's production cost advantage by producing for themselves as well as for their competitors.

invests in, are split into two categories: buyer-specific and non-specific—i.e. *general purpose*—assets. As explained above, the percentage of the 1 unit required for each buyer that is specific to that buyer, is the same as the extent to which that buyer's product is differentiated. The supplier adds the remaining, general purpose part for each buyer, across all the buyers she is matched to. We will assume that the supplier's *continuous* use of buyer-specific assets is subject to learning-by-doing, and that the supplier's *accumulation* of general purpose assets across the production for multiple buyers, is subject to scale economies. Both these relations are modeled using the following function:[12]

$$y = \max\left[0, 1 - \frac{1}{ax + 1 - a}\right],$$

which is represented graphically in Figure 1 for different values of $a$.

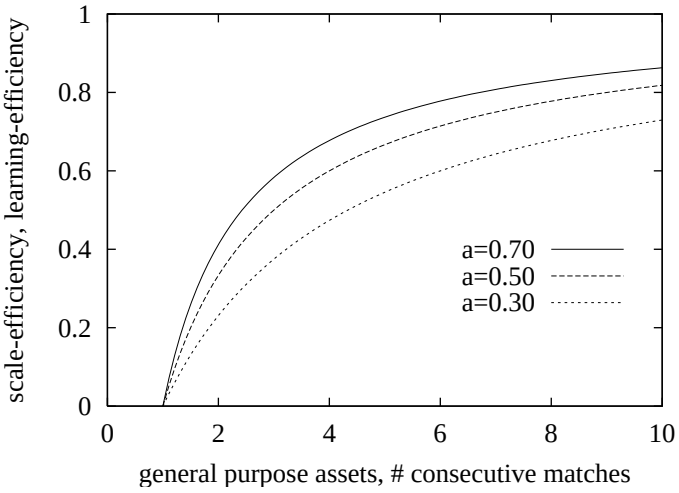


Figure 1: Efficiency of scale and of learning-by-doing.

If the $x$-axis measures a supplier's accumulation of general-purpose assets in the production for multiple buyers, then the $y$-axis gives the supplier's scale-efficiency in using those general-purpose assets. The number of general-purpose assets that supplier $j$ needs to produce for buyer $i$, is equal to $(1 - d_i)(1 - e_{s,j})$, where $d_i$ is the differentiation of buyer $i$'s products and $e_{s,j}$ is supplier $j$'s scale efficiency, which is the function value in Figure 1 of supplier $j$'s total number of general purpose assets, accumulated across all the buyers she is matched to. If the $x$-axis measures the number of consecutive matches between a supplier and a buyer, then the $y$-axis gives the

[12]Different values for the $a$-parameter may be used for the two functions. In the program, the parameter for the scale-efficiency function is `scaleFactor`, while the parameter for the learning-efficiency function is `learnFactor`.

supplier's buyer-specific efficiency in using assets, specific for that buyer.[13] The number of buyer-specific assets that a supplier $j$ needs to produce for a buyer $i$, is equal to $d_i(1 - e^i_{l,j})$, where $e^i_{l,j}$ is supplier $j$'s 'learning efficiency' (efficiency due to learning by doing) for buyer $i$, which is the function value in Figure 1, of the number of consecutive matches between buyer $i$ and supplier $j$.

The graph shows that a supplier can be more scale-efficient than a buyer producing for himself only if the scale at which she produces is larger than the maximum scale with which a buyer might produce for itself: the graph is positive only for more than 1 general purpose assets. Furhermore, a supplier's buyer-specific efficiency is 0 in their first transaction, and only starts to increase if the number of transactions is larger than 1, which implements TCE's *fundamental transformation*, according to which (Williamson 1981*b*, p. 1548),

> "[w]hat may have been (and commonly is) an effective large-numbers-bidding situation at the outset is sometimes *transformed* into a bilateral trading relation thereafter. This obtains if, despite the fact that large numbers of qualified bidders were prepared to enter competitive bids for the initial contract, the winning bidder realizes advantages over nonwinners at contract renewal intervals because nontrivial investments in durable specific assets are put in place (or otherwise accrue, say in a learning-by-doing fashion) during contract execution."

In the current model, the emphasis is put on the second option mentioned (between brackets). The relative effects of investments in durable specific assets vs. learning-by-doing advantages will be the subject of future work.

**In summary: profitability**$_{xy}$ The way profits are made, then, is that suppliers may reduce costs by generating efficiencies for buyers, while buyers may increase returns, when they sell more differentiated products. The profit that is made resulting from both partners' contributions, is shared equally between the buyer and the supplier involved.

### 5.1.2 Trust

TCE assumes potential opportunism. In the model, opportunism means that an agent may break a 'relation', i.e. a sequence of matches, without taking the partner into account. If agents know that their partner may be opportunistic, they can assign a probability to the event of their partner behaving opportunistically. This probability is 1 minus the probability that

[13] For now, the same function is used for both relations, although a learning curve is usually represented by a sigmoid function (cf. Simon and Blume 1994, p. 365).

the partner does not behave opportunistically, which we will call the agent's *trust* in the partner. Following Gulati (1995), we will assume trust to increase with the duration of a relation: as a relation lasts longer, one starts to take the partner's behavior for granted, and to assume the same behavior (i.e. commitment, rather than breaking the relation) for the future. In the model, this increase over time is implemented using a variation of the function presented in Figure 1. The addition is a base-level of trust:

$$y = b + (1 - b)\left(1 - \frac{1}{ax + 1 - a}\right),$$

where $b$ is the base-level of trust and $x$ is the number of consecutive matches the agents have been involved in. The parameter $a$ is again a different one than in the functions for scale- and learning-efficiency. In the program, this parameter is called `trustFactor`.

Technically, a base-level is desirable because if $\alpha = 1$, the exponent on trust is 0, and the base-level prevents trust from becoming 0 ($0^0$ is undefined, and makes the program crash). Theoretically, Hill (1990) also assumes that a certain proportion of the population will never be opportunistic, so that proportion may be taken as the agents' minimum probability-assessment that their partner will not be opportunistic; another interpretation is that this reflects a certain elementary decency in the population. Figure 2 shows the relation we assume between the past duration of a relation and agents' trust in each other (depending on an agent-specific value for the $a$-parameter in the function; i.e. $a = .5$).

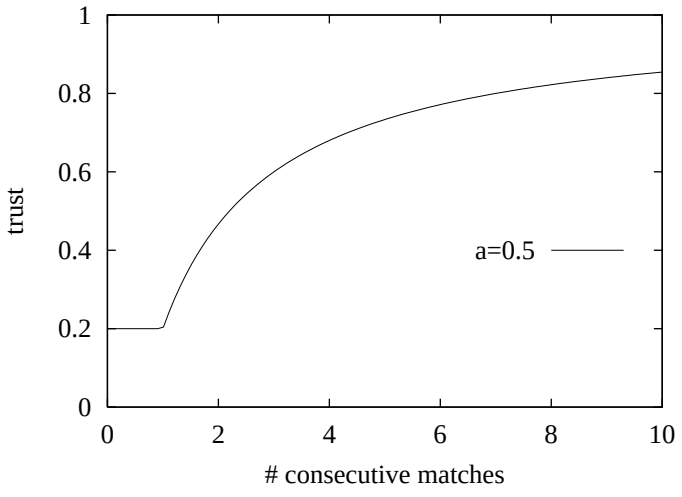


Figure 2: Trust.

A relation is broken if, during the matching, a buyer does not send any more requests to the supplier or he does, but the supplier rejects them. If

an agent $y$, involved in a relation with an agent $x$ 'breaks' their relation, then $x$'s trust in $y$ decreases; in effect, $x$'s trust drops by a certain factor ($< 1$) times the distance between the current level and the base-line level of trust; it stays there until the next time $x$ and $y$ are matched, after which is starts to increase again.

## 5.2 Implementation

### 5.2.1 Agent-based, object-oriented programming

The simulation was developed in the general-purpose, object-oriented programming language SIMULA (Birtwistle *et al.* 1973). The object-oriented paradigm is very well suited for agent-based modeling (see McFadzean and Tesfatsion 1996, Epstein and Axtell 1996), and for real-world modeling in general, which was the philosophy underlying the development of SIMULA as the first object-oriented language. Although the original language (SIMULA I) was a SIMUlation LAnguage, the second and final version, SIMULA 67 (nowadays just called SIMULA), is a general-purpose language, and the acronym now stands for SIMple Universal LAnguage. Object-oriented technology 'simulates' the real-world, which gives it several desirable properties. Object-oriented programs are modular; the modules are described in classes. These serve as 'templates' for the creation (instantiation) of objects, which represent actual objects in the real world. Classes consist of declarations of *data* (properties, attributes) and *methods* (behavior) that operate on those data. Subclasses may be defined that inherit the data and methods of the superclass, and may be re-defined or supplemented with data and methods specific for the subclass. Objects may also send messages to other objects. Object-oriented programming thus consists of specifying classes. If a program is run, the objects interact with each other by sending messages.

### 5.2.2 Simulation

The simulation proceeds as a sequence of discrete *timesteps* (see the pseudo-code listing for the simulation program in Table 1). Such a sequence is called a 'run'; each simulation experiment may be replicated several times (multiple runs), to reduce the influence of draws from random distributions on the results. During the step `initialize simulation` in Table 1, certain parameters are set for the simulation as a whole. The user is prompted to supply the number of buyers and suppliers, as well as the number of runs, and the number of timesteps in each run. The program's random number generator is seeded and finally, the agents are instantiated and given a number for identification. There is a general class `agent`, from which two subclasses, `buyerAgent` and `supplierAgent` are derived. The general class contains data and methods (called 'procedures' in SIMULA) that all agents

```
Begin simulation
initialize simulation;
For run:=1 Step 1 Until totalRuns Do:
{
    initialize agents;
    For timestep:=1 Step 1 Until totalTimesteps Do:
    {
        For agent:=1 Step 1 Until totalAgents Do:
        {
            choose a value for alpha;
            calculate scores;
            establish preference ranking;
        }
        matchAgents;
        For supplier:=1 Step 1 Until totalSuppliers Do:
            If matched Then produce and deliver;
        For buyer:=1 Step 1 Until totalBuyers Do:
            If not matched Then produce;
        For buyer:=1 Step 1 Until totalBuyers Do:
            sell;
        For agent:=1 Step 1 Until totalAgents Do:
            update;
    }
}
End simulation;
```

Table 1: Pseudo-code for the simulations.

have in common. They are 'inherited' by the two subclasses and supplemented with data and methods that are specific for buyers and suppliers, respectively (see Appendix A.2).

#### 5.2.3 Runs

The program then contains a set of nested for-loops, which control the required runs and, per run, the required timesteps: the statement

```
For run:=1 Step 1 Until totalRuns Do: {...}
```

lets the `{...}`-part be executed `totalRuns` times—i.e. as many times as the value of the variable `totalRuns`. The statement sets the variable `run` to 1 and increments it with 1 (`Step 1`) at the end of each loop. As long as the value of `run` is smaller than or equal to the value of `totalRuns`, the `{...}`-part is executed, otherwise the program continues after the closing `}`.

At the start of each run, each of the agents is initialized. For example, the agents' profits (from the previous run) are re-set to zero and the agents' trust in other agents is re-set.[14] After this agent-initialization, the actual simulation starts, consisting of a sequence of timesteps.

### 5.2.4 Timesteps

The matching algorithm is applied to the agents in *each* timestep, while the agents may adapt their preferences for other agents from each timestep to the next. In each timestep, *before* matching takes place, each agent chooses a value for $\alpha$ to calculate scores with, calculates scores, and ranks (potential) partners on the basis of these scores, using randomdraws to settle the ranking of alternatives with equal scores. Then, the agents are matched by the matching algorithm; suppliers that are matched to a buyer produce for and deliver to that buyer, while suppliers that are not matched do nothing; buyers that are not matched produce for themselves. Then, the buyers sell their products on the final-goods market—whether produced by their supplier or by themselves. Finally, at the *end* of each timestep, the agents do some updating on the basis of their experiences *during* the timestep.

The description of each of these events follows. What happens *before* the matching in any timestep (except the first, in which this is trivial)[15] is influenced by the events *after* the matching in the *previous* timestep. These latter events are therefore discussed first.

**The matching** Two 2-dimensional 'arrays' (matrices) are maintained in the program, in which connections (matches) before and after execution of the matching algorithm are stored. Right *before* each matching, the entries in the array of current connections are copied into the array of previous connections, and the array of current connections is cleared: the matching algorithm starts from scratch in each timestep and after it has finished, the resulting matches are stored in the array of current connections. Then, right *after* the matching, the entries in the two matrices are compared for each pair of agents, and the result of this comparison is classified as one of the following events in the life-cycle of a relation.[16]

**Start:** two agents *start* a relation if they are matched to each other in a certain timestep, while they were not matched in the previous timestep.

[14] A complete overview of all the different variables and parameters is given in Appendix A.3.

[15] Before the first matching, all buyers are the same for each supplier and vice versa. Because the matching algorithm needs strict preferences and random draws are used to break ties between alternatives with equal scores, all agents' preference rankings in the first timestep are random.

[16] If two agents are matched to each other neither before nor after the matching, then there is, of course, no event in a relation's life-cycle that this corresponds to.

**Continue:** two agents *continue* a relation if they are matched to each other in a certain timestep, while they were also matched to each other in the previous timestep. If a relation continues, the agents' trust in each other increases, as does the supplier's efficiency in using buyer-specific assets (if any).

**Break:** a relation *breaks* if, while two agents were matched to each other in the previous timestep, either the buyer does not send a request to the supplier or he does, but the supplier rejects the request. If a relation breaks, the trust of the agent who did not break the relation in the agent who did, decreases.

These events and the agents' perception, interpretation and evaluation of them may trigger reactions that may lead the agents to change their preference ranking, which, in the next timestep, may change the outcome of the matching and trigger the occurrence of further events.

**Production and trade** Producing one product requires (at most) one unit of assets—increasing efficiency decreases this amount. A buyer always produces with efficiency 0 if he chooses to make,[17] so he always needs 1 unit of assets to produce one product, costing 1 monetary unit. Buyer $i$'s prospects when making, then, are as depicted in Figure 3. A supplier, on

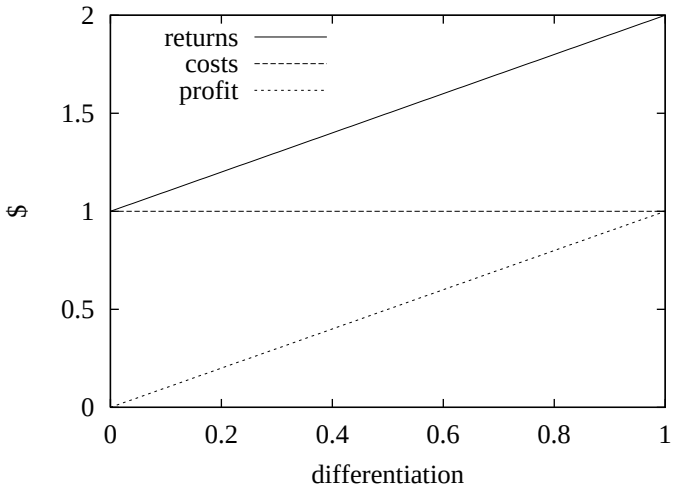


Figure 3: A buyer's returns, costs, and profit when making.

[17] A buyer is not allowed to produce for and supply to other buyers (his competitors), so he can not generate scale-economies. Learning-by-doing is also not possible for the buyer, because savings resulting from this are assumed to be related to the advantage due to the cognitive distance between the buyer and his supplier (cf. Nooteboom's (1992) *external economies of cognitive scope*, and the simulations by Péli and Nooteboom (1997)).

the other hand, may enjoy economies due to scale and due to learning by doing, as explained in Section 5.1.1. The difference between unity and the supplier's costs are the savings that the supplier generates, and the prices at which the supplier's production is traded with each of her buyers is such that these savings are *shared equally* between the buyer and the supplier. Finally, when the buyer sells his products, the price he receives is a function of the differentiation of his products. If the buyer has bought, rather than made, then like the supplier's savings, any returns resulting from differentiation are shared equally between the buyer and the supplier. This means that when a buyer $i$ buys from a supplier $j$, the profits they may make are equal:

$$\pi_i = \pi_j = 0.5d_i + 0.5d_i e^i_{l,j} + 0.5(1 - d_i)e_{s,j},$$

where $d_i$ is the differentiation of buyer $i$'s products, $e^i_{l,j}$ is supplier $j$'s learning efficiency for buyer $i$, and $e_{s,j}$ is supplier $j$'s scale efficiency. If the buyer buys, therefore, he faces the situation presented in Figure 4. The buyer's

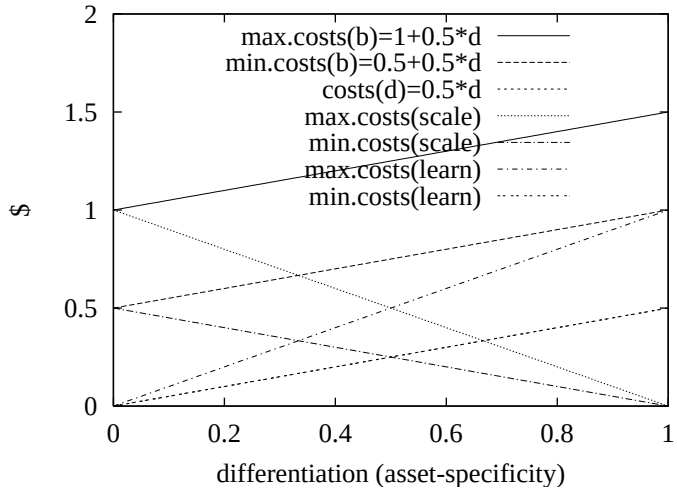


Figure 4: A buyer's costs when buying. The plot for 'costs(d)'—the buyer's sharing of his returns from differentiation with the supplier—coincides with the plot for 'min.costs(learn)'.

returns are the same as in Figure 3; his costs are unity minus half of the supplier's savings due to scale- and learning-efficiency, and he also shares half of his returns from product differentiation with the supplier ('costs(d)').

It follows, in Figure 5, that the buyer's profits when buying fall anywhere inbetween the lines 'min.profit(b)' and 'max.profit(b)'. Compared to the buyer's situation when making (the line 'profit(m)'), buying is more attractive—in terms of potentially attainable profits—when differentiation (and therefore asset specificity) is low, than when it is high, which is in line with transaction cost intuition.

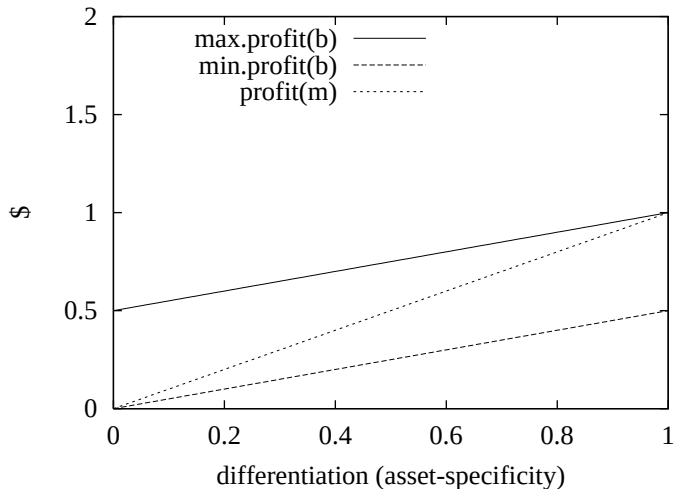


Figure 5: A buyer's profits when making or buying.

**Updating** An agent in a CAS is adaptive if "the actions of the agent in its environment can be assigned a value (performance, utility, payoff, fitness, or the like); and the agent behaves in such a way as to improve this value over time" (Holland and Miller 1991, p. 365; see also (Vriend 1995)). The adaptive character of the artificial agents in the model refers to the possibility for the agents to *change* the value they use for $\alpha$ from each timestep to the next, which leads to a change in the scores they assign to different agents and to a different preference-ranking. Each agent has several possible values for $\alpha \in [0, 1]$; the number is a parameter in the simulation. To each value, each agent assigns a *strength*,[18] which expresses the agent's confidence in the success of using that particular value; the various strenghts always add up to a constant $C$.

The strength of the value that was chosen for $\alpha$ at the start of a particular timestep (see below), is updated at the end of that timestep, on the basis of the agent's performance during that timestep, which is assumed to be related to the value of $\alpha$ used. Updating means that the agent adds the profit obtained during the timestep to the strength of the value used for $\alpha$. After this, the three strengths are renormalized to sum to $C$ again (see (Arthur 1993) for a discussion of this learning mechanism). This is done by multiplying each of them with the ratio $C/(C+\text{profit})$. At this point, as an output of the simulation, each agent $x$'s weighted average value for $\alpha_x$—the

[18] See (Arthur 1991, Arthur 1993, Kirman and Vriend 1998, Lane 1993) for discussions and applications of these so-called 'classifier systems' to models in economics; good general introductions are (Booker *et al.* 1989), (Goldberg 1989) and (Holland *et al.* 1986).

'profit-elasticity' of the scores that $x$ assigns—is calculated:

$$\sum_{\alpha_x=0,\ldots,1} \alpha_x \cdot \text{strength}(\alpha_x).$$

This indicates where $x$'s emphasis lies: because the value with the highest strength pulls the weighted average in its direction, the emphasis lies on low values for $\alpha$ if the weighted average $\alpha_x$ is low and vice versa.

**Choosing $\alpha$** The process of updating described in the previous paragraph concludes each timestep. The next timestep starts with each agent choosing a value to be used for $\alpha$ when calculating other agents' scores. The choice between the different possible values for $\alpha$ is probabilistic—a simple roulette wheel selection—with each value's selection probability equal to its relative strength, i.e. its strength divided by $C$.

**Calculating scores** As explained above, scores are Cobb-Douglas functions of profitability and trust:

$$\text{score}_{xy} = \text{profitability}_{xy}^{\alpha_x} \cdot \text{trust}_{xy}^{1-\alpha_x},$$

where $\text{score}_{xy}$ is the score $x$ assigns to $y$, $\text{profitability}_{xy}$ is the profit $x$ *may* make 'through' $y$, $\text{trust}_{xy}$ is $x$'s trust in $y$ and $\alpha_x \in [0, 1]$ is the importance $x$ attaches to $\text{profitability}_{xy}$ relative to $\text{trust}_{xy}$, i.e. the 'profit-elasticity' of the scores that $x$ assigns.

Before a matching, the agents determine other agents' scores on the basis of suppliers' scale-efficiency in the previous timestep. Only after the matching does it become clear to how many and which buyers each supplier is actually matched, and what the real extent of her scale-efficiency is. Expectations of the supplier's position on each buyer-specific learning curve, on the other hand, will already be accurate before the matching—assuming, of course, that the relation makes it through the matching.

# 6 Results: adaptive governance

Experiments were run with the parameters and variables as shown in the right-most column of Table 4 in Appendix A.3. The value for product differentiation was varied in 6 experiments, each of which was run for 250 timesteps and replicated 25 times: results are typically presented as averages over those 25 runs. Before going to the results, it is worthwhile to consider what may be expected from the simulations. The experimental variable 'differentiation' of the buyers' products is tied to the specificity of the assets that suppliers invest in to support their production for those buyers. Initially, therefore, the buyers are confronted with the score-differentials

| | $\alpha$ | | | | |
|---|---|---|---|---|---|
| $d$ | 0 | 0.25 | 0.5 | 0.75 | 1 |
| 0.25 | 0.50 | 0.23 | 0.06 | -0.05 | -0.13 |
| 0.35 | 0.40 | 0.17 | 0.01 | -0.10 | -0.18 |
| 0.45 | 0.30 | 0.11 | -0.04 | -0.15 | -0.23 |
| 0.55 | 0.20 | 0.03 | -0.10 | -0.20 | -0.28 |
| 0.65 | 0.10 | -0.04 | -0.16 | -0.25 | -0.33 |
| 0.75 | 0.00 | -0.12 | -0.22 | -0.30 | -0.38 |

Table 2: Difference between suppliers' initial scores and a buyer's own score ($= d$) for different values of differentiation and of the buyer's $\alpha$.

given in Table 2. The values in Table 2 are calculated as follows. The score that a buyer $i$ assigns to a supplier $j$, is

$$\text{score}_{i,j} = (0.5d_i + 0.5d_i e^i_{l,j} + 0.5(1 - d_i)e_{s,j})^\alpha \cdot t_i^{j(1-\alpha)},$$

with the supplier's initial learning-efficiency for buyer $i$, $e^i_{l,j} = 0$, the supplier's initial efficiency of scale, $e_{s,j} = 0$ and buyer $i$'s initial trust in supplier $j$, $t_i^j = 0.75$. The score that buyer $i$ assigns to himself is equal to $d_i$, because that is his profit when he makes and he uses $\alpha = 1$ to calculate his own score. The values in the table give the difference between these two; initially (in the first timestep), these values are the same for all buyers. As differentiation increases, the number of distinct values for $\alpha$ that yield a net score-advantage for suppliers—which they need for buyers to consider them acceptable—decreases. If $d = 0.75$, there is no value for $\alpha$ that gives suppliers a net advantage so we may expect no outsourcing at all in that case. Notice that for *any* $d < 0.75$, no matter how much smaller, the suppliers do have a net advantage, which, furthermore, *if* matches do occur, increases over time with suppliers' increasing learning efficiency and also when suppliers are matched to more than 1 buyer. The situation in Table 2, therefore, is likely to shift in favor of suppliers as time progresses. In general, then, we would expect more making (and less buying) when differentiation increases.

The proportion of economic activity under hierarchichal (as opposed to market) governance in the different experiments is shown in Figure 6. This shows that, as expected, the proportion made is higher when differentiation is high than when it is low, and if $d = 0.75$, nothing is bought; the buyers make everything themselves (the plot for $d = 0.75$ coincides with the top border of the graph). Notice however, that in all experiments, the proportion made decreases during approximately the first 20 timesteps, after which it increases and more strongly so, when $d$ is higher.

The corresponding plot for the buyers' average *normalized* profits is shown in Figure 7—notice that the $y$-axis was re-scaled to 0.5–1 to make the results clearer. Again, the plot for the case $d = 0.75$ coincides with the

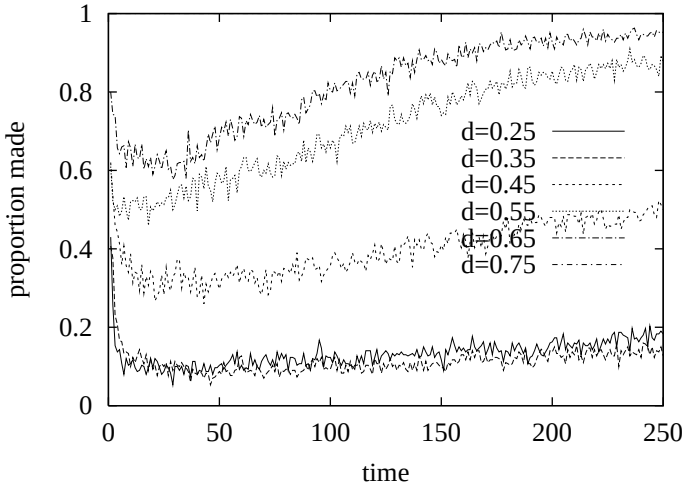


Figure 6: Proportion 'made' (as opposed to 'bought').

top border of the graph. This normalized profit is the buyers' profit divided by the maximum attainable profit, which is the profit they would make in a relation with a supplier with maximum scale- and learning-efficiency.[19] Figure 7 shows, first of all, that the initial decrease in 'proportion made'

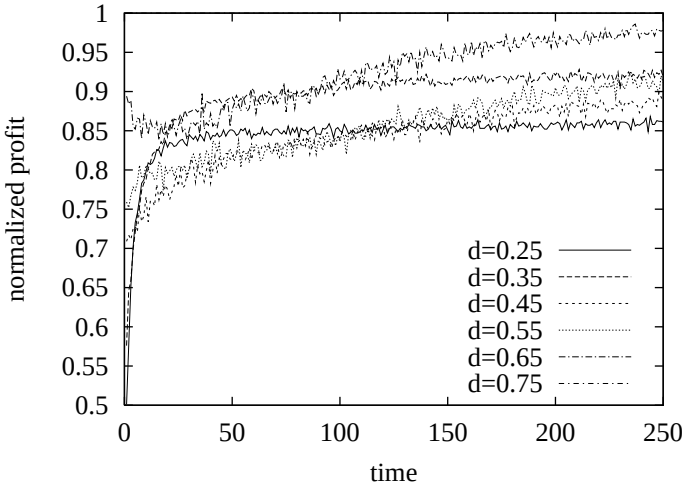


Figure 7: Buyers' normalized profits.

(see Figure 6) is 'good' for the buyers when differentiation is low (their normalized profit increases during this initial period), but 'bad' when differ-

[19]This is corrected for the fact that the suppliers' scale-efficiency is limited because their acceptance quotum is set to 3; if $a_s$ is unlimited, the system quickly settles in a state where all buyers buy from the same supplier.

entiation is high, since the theoretically most appropriate choice is to make when differentiation is high. Eventually, this is also what the agents learn. Furthermore, in several of the experiments, the agents are performing poorer than they could be. This is because 'perfomance as it could be' is based on profit made in a relation with a supplier with maximum scale- and learning-efficiency. Since each supplier can have a maximum of 3 buyers ($a_s = 3$), this requires that the 12 buyers together buy from only 4 suppliers. That this network configuration does nog always emerge, is shown in Figure 8, which depicts the buyers' average normalized profits in each of the 25 individual runs of experiment $d = 0.35$. In this experiment, there are three levels at

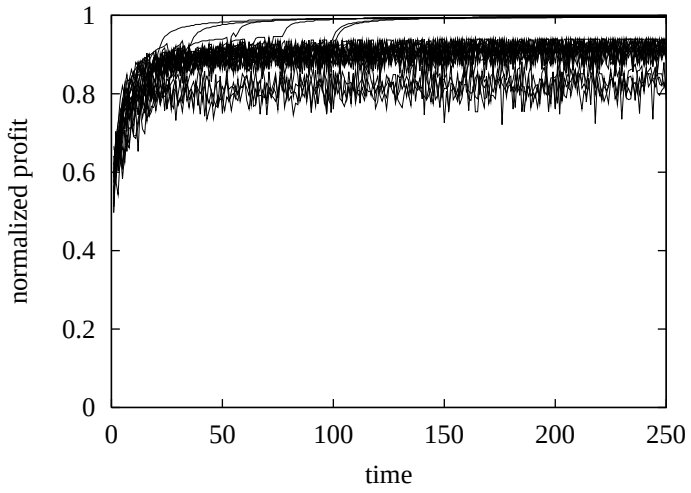


Figure 8: Buyers' normalized profits in 25 runs of experiment $d = 0.35$.

which average profits 'stabilize'; almost 1 (6 runs), and approximately 0.9 (15 runs) and 0.8 (4 runs). The first of these levels corresponds to the situation where the 12 buyers buy from 4 suppliers (with their maximum of 3 buyers each) and no buyer makes anything. The second level corresponds to the situation where 9 buyers are consistently matched to 3 suppliers and the other three buyers are either making or buying, but not all three from the same supplier at the same time. Also, there is much switching between suppliers in this case, so these three buyers form no long-lasting relations. The final level (0.8) corresponds to the situation when even more buyers are not consistently buying from the same supplier who is matched to her maximum number of buyers. If the simulation is re-run with 12 buyers but only 4 suppliers, most of the runs quickly lock in to the level 1 described above.

The corresponding weighted average $\alpha$ for the buyers, averaged over all buyers, is displayed in Figure 9. The weighted average $\alpha$ goes up in all experiments and more strongly when differentiation is higher. When $d = 0.75$,

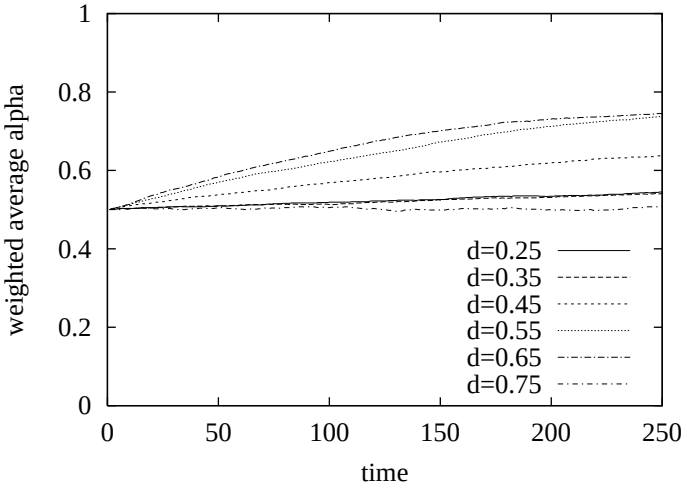


Figure 9: Buyers' weighted average $\alpha$.

there is hardly any effect on this variable: because there is no outsourcing at all in this case, the profit that is made is the same no matter which value was used for $\alpha$, so no one value is better than the rest in this sense. Because the buyers do attain maximum normalized profit when differentiation is higher, the higher weighted average $\alpha$ that emerges in that case can be called optimal. However, the buyers do not perform as well when differentiation is low, which implies that the weighted average $\alpha$ may not be optimal (too high or too low), although the agents may not be aware of this. The optimum may be out of reach of the path-dependent process of interaction and learning that unfolds among the agents. When looking at the 25 individual runs of the experiment $d = 0.25$, it appears that relatively high profits are correlated with no 'making' (only buying), and a relatively low weighted average $\alpha$! Further research will be done to investigate this further.

# 7 Conclusion

This study was motivated by the observation that, while the main hypothesis of transaction cost economics is that agents are able to align organizational form with the attributes of transactions in a discriminating, (transaction cost) economic way, the agents' bounded rationality may *prevent* them from performing this alignment successfully, i.e. economically. An agent-based computer simulation model was developed and implemented to generate alternative propositions about which organizational forms boundedly rational, adaptive agents learn to use. It was shown that under some circumstances, agents are indeed unable to optimize successfully, because the network they form, along with other agents, may not evolve to an optimal configuration.

The agents learn 'individually', and in this case (as in many others), a population of agents pursuing their own self-interest does not—by any invisible hand—lead to a globally optimal outcome; consequently, some of the gains from trade are not reaped. This may be interpreted as an example of how "the analysis cannot be confined to what happens within a single firm" (Coase 1995, p. 245).

An often-used technique in agent-based computational models is the genetic algorithm (see (Holland and Miller 1991) for a discussion and (Miller 1996, Tesfatsion 1999) for examples). A genetic algorithm (GA) is essentially a computational search heuristic that simulates evolutionary processes of selection and reproduction, operating on a population of potential solutions in its search for the optimal solution. In ACE-models, it operates on individual agents' behavioral rules, and genetic operators are assumed to model cultural, rather than genetic, transmission of ideas and behaviors. Using a GA, the optimal distribution of economic activity may have been found in the current experiments, but it is my opinion that a GA is not an accurate model of the learning of individual, boundedly rational agents that we have tried to capture in the current application (cf. Klos 1999).

The appropriateness of GA's as a model of learning in economics is the subject of much debate (Brenner 1998, Chattoe 1998, Riechmann 1999); explicit comparisons between social learning, modeled using a GA, and individual learning are reported in (Klos 1999) and (Vriend 1998), among others. There, it was shown that outcomes differ significantly between the two approaches. The GA comes out on top, but, in my opinion, at the expense of some very strong modeling assumptions—that many authors implicitely impose—about the extent of the agents' perception and computational capacity (see Klos 1999, for a more detailed discussion). Tesfatsion (1999, p. 13–14, emphasis added) *is* aware of this, given her observation that

> "[a]n important caution is in order here, however. Given the extent of information currently allowed to agents during the evolution step—i.e., knowledge of the complete strategies of all other agents of the same type, whether expressed in interactions or not—the evolution step is more appropriately interpreted as an iterative stochastic search algorithm for determining potential equilibrium strategy configurations rather than as a cultural transmission mechanism per se. The resulting earnings outcomes will be used in subsequent work as benchmarks against which to assess the effectiveness of *more realistically modelled* cultural transmission mechanisms".

According to Vriend (1998, p. 11), also, "the computational modeling choice between individual and social learning algorithms should be made more carefully, since there may be significant implications for the outcomes generated". As in (Klos 1999), I do not want to say that GA's should not be

used anymore, but the choice of using them should be made with caution and they should be used for the right purposes. In the current paper, for example, if in some circumstances agents are unable to organize optimally, it would be interesting to find out what 'optimal' would be in those settings, what is keeping the agents from attaining it, and how they might go about reaching that optimum. A GA might very well be used to find out what the optimum is, but that result should not be interpreted as the outcome of a process of learning. After all, it was because we question TCE's assumption that agents are able to align optimally that the current project was undertaken, so we shouldn't then use an optimization algorithm to model boundedly rational agents' adaptive learning.

# A Appendix

## A.1 Matching example

For an example of the operation of the matching algorithm, consider Table 3, which lists randomly generated preference rankings of 5 buyers over 5 suppliers and vice versa. In addition, the buyers were placed at randomly

| | supplier | | | | |
|---|---|---|---|---|---|
| buyer | 1 | 2 | 3 | 4 | 5 |
| 1 | 4,2 | 5,1 | 2,4 | 1,1 | 3,1 |
| 2 | 1,4 | -,2 | -,1 | -,2 | -,5 |
| 3 | 3,1 | -,4 | 4,2 | 2,4 | 1,3 |
| 4 | -,3 | -,5 | -,5 | 1,3 | 2,2 |
| 5 | -,5 | -,3 | -,3 | -,5 | -,4 |

Table 3: Example preference-rankings in Gale and Shapley's (1962) format. Buyer 1 ranks supplier 4 first, 3 second, 5 third, etc. Supplier 1 ranks buyer 3 first, 1 second, 4 third, etc. Buyer 2 has only one acceptable supplier (1); a '-' means 'unacceptable'.

generated positions on their own rankings (expressing their tolerance level) and suppliers whose ranking was not higher than the buyer's own ranking are not acceptable and therefore not listed.

If all agents are allowed only one partner ($o_b = a_s = 1$), the algorithm produces the following steps.

1. Buyers 1, 2, 3 and 4 send requests to their most preferred suppliers, i.e. 4, 1, 5 and 4, respectively. The suppliers that receive only one request accept those provisionally, while supplier 4 rejects the request from buyer 4 and provisionally accepts the request from buyer 1.

2. Buyer 4 sends a request to its next most preferred supplier, 5, who accepts buyer 4's request and rejects buyer 3's already provisionally accepted request, because supplier 5 prefers buyer 4 to buyer 3.

3. Buyer 3 now sends a request to its next most preferred supplier, 4, which request is rejected, because supplier 4 prefers its already accepted buyer (1) to buyer 3.

4. Buyer 3 now sends a request to the next supplier on its list, which is supplier 1, who accepts that request and rejects buyer 2's request, which it had previously accepted provisionally.

5. Buyer 2 has no more acceptable suppliers so no buyer sends another request, which stops the algorithm.

Buyers 1, 3 and 4 are now matched with suppliers 4, 1 and 5, respectively. The algorithm is also able to handle cases where $o_b$ and/or $a_s$ are greater than 1. For example, the reader may verify that buyers 1, 2, 3 and 4 will be matched to suppliers (3 and 4), (1), (1 and 5) and (4 and 5), respectively, when $o_b = a_s = 2$.

## A.2 Agent-specification

The program consists of a main loop (see Table 1), a procedure called `matchAgents`, and the declaration of several classes. The class `agent` contains the following procedures:

`setAlpha` This is the procedure that chooses a value to be used for $\alpha$. A random number between 0 and 1 is drawn like a roulette wheel being spun. The wheel is divided like a pie in as many parts as there are possible values for $\alpha$, with the size of each part proportional to the relative strength of the associated value for $\alpha$.

`updateWeights(alphaUsed,payoff)` This procedure is called for updating the strengths, associated with the different possible values for alpha. The parameter `alphaUsed` is the value that was used for $\alpha$ and of which the strength needs to be updated. This is done by adding the value of `payoff` to the strength. Then, each strength is multiplied with the ratio $C/(C+$`payoff`$)$, to ensure that they add up to C again.

From the class `agent`, the classes `buyerAgent` and `supplierAgent` are derived. These subclasses inherit all data and methods from the class `agent`. In addition, data and methods are declared specifically for the two subclasses. The subclass `buyerAgent` contains the following procedures:

`calculateSupplierScores` This procedure calculate scores of suppliers and self as Cobb-Douglas functions of profitability and trust, as described above in Section 5.1. If the buyer's value for $\alpha$ is 0, then the supplier's score is simply equal to the buyer's trust in the supplier. The buyer calculates his own score using efficiency = 0, trust is 1, and $\alpha = 1$.

`buyerProcess` If not matched then make; in any case, sell. The suppliers' equivalent process `supplierProcess` is executed before the buyers', so if a buyer is matched to a supplier, that supplier will already have produced for him.

`increaseTrust(subject)` This increases the buyer's trust in `subject` on the basis of the number of previous times they have been matched.

`decreaseTrust(subject)` This decreases the buyer's trust in `subject`.

Besides the data and methods inherited from the class `agent`, the subclass `supplierAgent` contains the following procedures:

`calculateBuyerScores` This procedure calculates the scores the supplier assigns to each buyer. As in the buyer's equivalent procedure, if a buyer's profitability as well as the supplier's $\alpha$ are 0, the supplier's trust is used as the buyer's score.

`determineScaleEfficiency` The supplier adds the general purpose assets required for producing for all the buyers she is matched to and calculates the scale-function value of this number. This is the supplier's scale efficiency.

`climbLearningCurve(subject)` On the basis of the number of times they have been matched before, the supplier calculates her efficiency in using `subject`-specific assets.

`increaseTrust(subject)` This procedure increases the supplier's trust in `subject` on the basis of the number of previous times they have been matched.

`decreaseTrust(subject)` This procedure decreases the supplier's trust in `subject`.

`produceFor(subject)` Based on buyer `subject`'s differentiation and the supplier's (general) scale- and `subject`-specific efficiency, the supplier acquires the required assets and produces for `subject`.

`supplierProcess` Looking at each buyer in turn, if the supplier is matched to that buyer, it produces for that buyer (see previous procedure).

### A.3 Parameters and variables

This appendix gives a complete overview of all the parameters and variables, used in the simulation; see Table 4.

| | param./var. | value range | value used |
|---|---|---|---|
| **general** | number of buyers, $B$ | $\{1, 2, \ldots\}$ | 12 |
| | number of suppliers, $S$ | $\{1, 2, \ldots\}$ | 12 |
| | number of runs | $\{1, 2, \ldots\}$ | 25 |
| | number of timesteps | $\{1, 2, \ldots\}$ | 250 |
| **per buyer** | differentiation | $[0, 1]$ | $\{0.25, 0.35, \ldots, 0.75\}$ |
| | $o_b$ | $\{1, 2, \ldots, S\}$ | 1 |
| | number of values for $\alpha$ | $\{2, 3, \ldots\}$ | 5 |
| | $C$ | $\langle 0, \ldots \rangle$ | 20 |
| | baseTrust | $\langle 0, 1]$ | 0.3 |
| | initTrust(subject) | $\langle 0, 1]$ | 0.75 |
| | trustFactor | $[0, 1]$ | 0.5 |
| **per supplier** | $a_s$ | $\{1, \ldots, B\}$ | 3 |
| | scaleFactor | $[0, 1]$ | 0.5 |
| | learnFactor | $[0, 1]$ | 0.5 |
| | number of values for $\alpha$ | $\{2, 3, \ldots\}$ | 5 |
| | $C$ | $\langle 0, \ldots \rangle$ | 20 |
| | baseTrust | $\langle 0, 1]$ | 0.3 |
| | initTrust(subject) | $\langle 0, 1]$ | 0.75 |
| | trustFactor | $[0, 1]$ | 0.5 |

Table 4: Parameters and variables in the simulation.

## References


Albin, Peter S. and Duncan K. Foley (1992). Decentralized, dispersed exchange without an auctioneer: A simulation study. *Journal of Economic Behavior and Organization* **18**(1), 27–51.

Anderson, Simon P., André de Palma and Jacques-François Thisse (1992). *Discrete Choice Theory of Product Differentiation*. The MIT Press. Cambridge, MA.

Arthur, W. Brian (1991). Designing economic agents that act like human agents: A behavioral approach to bounded rationality. *American Economic Review* **81**(2), 353–359.

Arthur, W. Brian (1993). On designing economic agents that behave like human agents. *Journal of Evolutionary Economics* **3**(1), 1–22.

Arthur, W. Brian, John H. Holland, Blake LeBaron, Richard Palmer and Paul Tayler (1997). Asset pricing under endogenous expectations in an artificial stock market. In: *The Economy as an Evolving Complex System, II* (W. Brian Arthur, Steven N. Durlauf and David A. Lane, Eds.). Vol. XXVII of *Santa Fe Institute Studies in the Sciences of Complexity Proceedings*. pp. 15–44. Addison-Wesley. Reading, MA.

Axtell, Robert L. (1999). The emergence of firms in a population of agents: Local increasing returns, unstable Nash equilibria, and power law size distributions. Working paper. Brookings Institution.

Birtwistle, Graham M., Ole-Johan Dahl, Bjørn Myhrhaug and Kristen Nygaard (1973). *SIMULA Begin*. Studentlitteratur. Lund, Sweden.

Booker, Lashon B., David E. Goldberg and John H. Holland (1989). Classifier systems and genetic algorithms. *Artificial Intelligence* **40**(1–3), 235–282.

Brenner, Thomas (1998). Can evolutionary algorithms describe learning processes?. *Journal of Evolutionary Economics* **8**(3), 271–283.

Chattoe, Edmund (1998). Just how (un)realistic are evolutionary algorithms as representations of social processes?. *Journal of Artificial Societies and Social Simulation* **1**(3),
`http://www.soc.surrey.ac.uk/JASSS/1/3/2.html`.

Coase, Ronald H. (1937). The nature of the firm. *Economica NS* **4**(16), 386–405.

Coase, Ronald H. (1995). My evolution as an economist. In: *Lives of the Laureates* (William Breit and Roger W. Spencer, Eds.). pp. 227–249. The MIT Press. Cambridge, MA.

Coase, Ronald H. (1998). The new institutional economics. *American Economic Review* **88**(2), 72–74.

Epstein, Joshua M. and Robert L. Axtell (1996). *Growing Artificial Societies: Social Science from the Bottom Up.* Brookings Institution Press/The MIT Press. Washington, DC/Cambridge, MA.

Gale, David and Lloyd S. Shapley (1962). College admissions and the stability of marriage. *American Mathematical Monthly* **69**(January), 9–15.

Glance, Natalie S. and Bernardo A. Huberman (1994). Social dilemmas and fluid organizations. In: *Computational Organization Theory* (Kathleen M. Carley and Michael J. Prietula, Eds.). pp. 217–239. Lawrence Erlbaum. Hillsdale, NJ.

Goldberg, David E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley. Reading, MA.

Gulati, Ranjay (1995). Does familiarity breed trust? The implications of repeated ties for contractual choice in alliances. *Academy of Management Journal* **38**(1), 85–112.

Hill, Charles W.L. (1990). Cooperation, opportunism, and the invisible hand: Implications for transaction cost theory. *Academy of Management Review* **15**(3), 500–513.

Holland, John H. (1992). Complex adaptive systems. *Daedalus* **121**(1), 17–30.

Holland, John H. and John H. Miller (1991). Artificial adaptive agents in economic theory. *American Economic Review* **81**(2), 365–370.

Holland, John H., Keith J. Holyoak, Richard E. Nisbett and Paul R. Thagard (1986). *Induction: Processes of Inference, Learning, and Discovery.* The MIT Press. Cambridge, MA.

Kirman, Alan P. and Nicolaas J. Vriend (1998). Evolving market structure: A model of price dispersion and loyalty. Paper for the $3^{rd}$ Workshop on Economics with Heterogenous Interacting Agents. Ancona, Italy.
`http://www.econ.unian.it/dipartimento/siec/hia98/papers/Vrie_kir.pdf`.

Klos, Tomas B. (1999). Decentralized interaction and co-adaptation in the repeated prisoner's dilemma. *Computational and Mathematical Organization Theory* **5**(2), 147–165.

Kollman, Ken, John H. Miller and Scott E. Page (1992). Adaptive parties in spatial elections. *American Political Science Review* **86**(4), 929–937.

Lane, David A. (1993). Artificial worlds and economics, part II. *Journal of Evolutionary Economics* **3**(3), 177–197.

McFadzean, David and Leigh S. Tesfatsion (1996). A C++ platform for the evolution of trade networks. Economic Report 39. Iowa State University.
`http://www.econ.iastate.edu/tesfatsi/platroot.ps`.

Miller, John H. (1996). The coevolution of automata in the repeated prisoner's dilemma. *Journal of Economic Behavior and Organization* **29**(1), 87–112.

Miller, John H. and Peter F. Stadler (1998). The dynamics of locally adaptive parties under spatial voting. *Journal of Economic Dynamics and Control* **23**(2), 171–189.

Nooteboom, Bart (1992). Towards a dynamic theory of transactions. *Journal of Evolutionary Economics* **2**(3), 281–299.

Olson, Jr., Mancur (1965). *The Logic of Collective Action: Public Goods and the Theory of Groups*. Harvard University Press. Cambridge, MA.

Péli, Gábor L. and Bart Nooteboom (1997). Simulation of learning in supply partnerships. *Computational and Mathematical Organization Theory* **3**(1), 43–66.

Riechmann, Thomas (1999). Learning and behavioral stability: An economic interpretation of genetic algorithms. *Journal of Evolutionary Economics* **9**(2), 225–242.

Roth, Alvin E. and Marilda A. Oliveira Sotomayor (1990). *Two-sided Matching: A Study in Game-theoretic Modeling and Analysis*. Vol. 18 of *Econometric Society Monographs*. Cambridge University Press. Cambridge (UK).

Simon, Carl P. and Lawrence E. Blume (1994). *Mathematics for Economists*. Norton. New York.

Stanley, E. Ann, Dan Ashlock and Leigh S. Tesfatsion (1994). Iterated prisoner's dilemma with choice and refusal of partners. In: *Artificial Life III* (Chris G. Langton, Ed.). Vol. XVII of *SFI Studies in the Sciences of Complexity*. Addison-Wesley. Reading, MA. pp. 131–175.

Tesfatsion, Leigh S. (1996). A trade network game with endogenous partner selection. Economic Report 36. Iowa State University. Ames, IA.

Tesfatsion, Leigh S. (1999). Market power effects in evolutionary labor markets with adaptive search. Economic Report 48. Iowa State University.

Vriend, Nicolaas J. (1995). Self-organization of markets: An example of a computational approach. *Computational Economics* **8**(3), 205–232.

Vriend, Nicolaas J. (1996). A model of market-making. Economics Working Paper 184. Universitat Pompeu Fabra. Barcelona.

Vriend, Nicolaas J. (1998). An illustration of the essential difference between individual and social learning, and its consequences for computational analyses. Working Paper 387. Queen Mary and Westfield College, University of London. London.
`http://www.econ.qmw.ac.uk/papers/wp387.pdf`.

Weisbuch, Gérard, Alan P. Kirman and Dorothea K. Herreiner (1998). Market organisation and trading relationships. Working Paper.

Williamson, Oliver E. (1975). *Markets and Hierarchies: Analysis and Antitrust Implications*. The Free Press. New York.

Williamson, Oliver E. (1979). Transaction cost economics: The governance of contractual relations. *Journal of Law and Economics* **22**(2), 233–261.

Williamson, Oliver E. (1981*a*). The economics of organization: The transaction cost approach. *American Journal of Sociology* **87**(3), 548–577.

Williamson, Oliver E. (1981*b*). The modern corporation: Origins, evolution, attributes. *Journal of Economic Literature* **19**(4), 1537–1568.

Williamson, Oliver E. (1985). *The Economic Institutions of Capitalism: Firms, Markets, Relational Contracting*. The Free Press. New York.

Williamson, Oliver E. (1991). Comparative economic organization: The analysis of discrete structural alternatives. *Administrative Science Quarterly* **36**(2), 269–296.