LEARNING BY MIGRATING: A COMPUTATIONAL STUDY OF DIVERSITY AND TEAM-LEVEL DECISION-MAKING

Russell C Thomas[1] and John S Gero[2]
[1]George Mason University
[2]George Mason University and University of North Carolina, Charlotte

Abstract
How does previous experience and learning influence a team's ability to successfully agree on a system architecture, team roles and responsibilities, and design method? Migration of team members leads to diversity in past experiences and beliefs, which might have a positive or negative affect on team decision-making. Using computational modeling of self-managed teams across multiple project life cycles, we perform controlled experiments to evaluate performance and decision-making patterns of migrating vs. non-migrating teams. We find that there is no difference in mean performance, indicating that neither approach is intrinsically better. However, statistical tests of paired trials show a meaningful an advantage for migrating (diverse) teams. Examining patterns of decision-making over time reveal that migrating (diverse) teams explore a wider range of team-level decisions, which makes them more adaptable in specific circumstances.

1 INTRODUCTION

Many approaches to improving engineering design productivity depend on making changes to team structure, processes, or methods. In this research we examine the influence of team diversity on decision-making, specifically decisions on system design, roles and responsibilities, and design methods.

For example, consider an electrical appliance design team that is making the transition from traditional design (a.k.a. "Traditionalist") to Design for Sustainability (a.k.a. "Green") (Wever, Kuijk and Boks, 2008; Verhulst and Boks, 2012). In traditional design, team goals include product performance and functionality, unit cost, styling, production volumes, and design project schedule. Design for Sustainability introduces lifecycle goals such as reducing electricity consumed over the life of the appliance and reducing or eliminating end-of-life waste. To meet these goals, designers might introduce features that create new dependencies between subsystems, which in turn might require more or different collaboration between engineers on those subsystems. These new dependencies might also make it more difficult to meet the other goals. For these reasons Design for Sustainability is often promoted through new design processes and methods (Bhamra, Lilley and Tang, 2011).

Individual team members also influence the way a team works through their past experiences, beliefs, expectations, and patterns of social interaction. This is called 'situated social cognition' (Smith and Gero, 2005) or 'distributed cognition' (Nardi, 1996). When designers work together over an extended period, they tend to internalize a common way of working and thinking, though not necessarily identical. (For a survey of small group coordination, management, and work, see Chidambaram and Bostrom (1996)). Of particular importance for our research is the Time, Interaction, and Performance (TIP) theory of groups (McGrath, 1991), which emphasizes the temporal processes involving task performance by individuals and group interaction among individuals.

Organization context is also important. Large organizations often have many design teams with different ways of working, and especially if they are in separate business units or divisions. What happens when designers migrate to new teams with very different ways of working and thinking? In our example above: What happens when one or more "Traditionalist" designers joins a "Green" team, or vice versa? Generally, we can imagine three possible outcomes. First, the design team might be *negatively affected* if there is excessive conflict or disruption due to the new members and their different ways of working. Second, the design team might be *positively affected* if there is synergy or constructive interaction between the established ways of working and the new member's ways. Third, there might be *no significant change*, especially if the new member is quickly acculturated and otherwise does not affect how the team works or thinks.

Productivity and teamwork effects manifest in many ways, and there is extensive literature on how team diversity affects cohesion, conflict resolution, gender issues, etc. For example, a number of researchers have studied the effects of membership changes and dissent (Akgün and Lynn, 2002; Dreu, 2002; Hansen and Levine, 2009, Hirst, 2009; Rink et al., 2013; van der Haar et al., 2015). Knowledge acquisition and knowledge sharing in small teams has been studied by Walz, Elam, and Curtis (1993) and Reagans, Argote, and Brooks (2005). Cross (2004) organizes a review of research on team design and project lifecycles into three distinct categories: 1) problem formulation, 2) solution generation, and 3) process strategies. Our work focuses on problem formulation and process strategies, with solution generation modelled abstractly as an evolutionary search process (Metcalfe 2007). Notably, there has been only a little research (e.g. Singh, Dong and Gero, 2011) on how team heterogeneity affects design and process decisions. The scope of this research needs to include cognitive process of 'reflexivity' where agents perceive and reason about their own behaviour, including their cognitive processes, i.e. meta-cognition (Schippers, Den Hartog and Koopman, 2007). (Note: "reflexivity" is the term that has been adopted in the literature even though "reflectivity" is more accurate etymologically.)

This leads to our research question: *How does previous experience and learning influence a team's ability to successfully agree on a system design, team roles and responsibilities, and design method?*

The paper is organized as follows. Section 2 describes our research method – computational modelling, simulation, and controlled experiments. Section 3 presents an overview of our computational model and simulations. Section 4 presents experiment results with analysis. The paper closes with Section 5, a discussion of findings, implications, limitations, and extensions.

2 METHOD

We use agent-based modeling (ABM) as our general approach to computational modeling, along with a cognitive architecture for Designer agents. In ABM, behavior of the system as a whole is governed by interactions between relatively autonomous agents who act locally and who learn from experience. We have chosen to model a design task environment at a fairly abstract level because we are primarily concerned with how Designers reflect on their task-level experience, form beliefs, and use those beliefs in collective decision-making.

This approach has two advantages. First, it allows us to perform randomized controlled experiments that are similar to laboratory human subject experiments. Second, computational experiments are not limited in scale, scope, or measurement accuracy, compared to human subject experiments. Studying two or more design teams over multiple project lifecycles would not be feasible in a laboratory setting.

We model Designer cognition using the ECHO model (Thagard, 1989). ECHO is a connectionist computational model of the Theory of Explanatory Coherence (Thagard 2000), which is a general theory of how people make sense of their world and each other. The theory has been applied to reasoning by scientists and scientific theories, including conceptual revolutions (Thagard, 1992; 2012) and has been supported by empirical research (Schank and Ranney, 1991). ECHO is a limited model of cognition because it omits perception, affect, and many other capabilities. The advantage of a limited model is that it simplifies implementation, testing, and experimental validation.

We model design work as an evolutionary search process over a rugged landscape of possible design decisions (Metcalfe, 2007). This is a "black box" approach that abstracts away from the conceptual details of design work, and frames design as bounded-rational process of making interdependent design decisions. This approach has been used to study organization decision-making (Ethiraj and Levinthal, 2004). This modeling approach creates an artificial world that glosses over many real-world phenomena and processes, including creativity, conflict resolution, and the role of domain knowledge. On the plus side, this modeling approach is computationally difficult for our Designer agents, individually and as teams, and therefore it is sufficient to provide the environment in which to study agent reasoning and collective decision-making. It also has an experimental advantage in that it is straightforward to randomly generate performance landscapes using a few parameters.

3 COMPUTATIONAL MODEL

This section provides an overview of our computational model, emphasizing the architecture and functional significance of key aspects.

3.1 Project Lifecycle

A project lifecycle (Figure 1) consists of all phases of design, starting with team formation and ending when a project is completed. When a team forms, Designers engage in a collective decision-making process to decide on 1) product architecture, 2) design process, and 3) assignment of roles. Assuming they can make a consensus decision, each Designer then performs design tasks until the design goal is reached or until the time limit for the project is reached. Each Designer then collects evidence from their experience and updates their mental model of the design process. At the end, some Designers migrate to other teams and the lifecycle repeats.

3.2 Computational Architecture

The computational architecture consists of three entities: Teams, Designers, and the Performance Landscape.

3.3 Teams

Teams are collections of Designer agents who perform a single design project lifecycle. There are four teams that operate simultaneously and synchronously, but are otherwise independent of each other. The total number of teams is not critical to the experiment results, other than to provide diversity of experiences for Designers.

In Pre-design, Designers engage in collective decision-making process in order to decide on system architecture (see Section 4.5, below), design process, and role assignments.

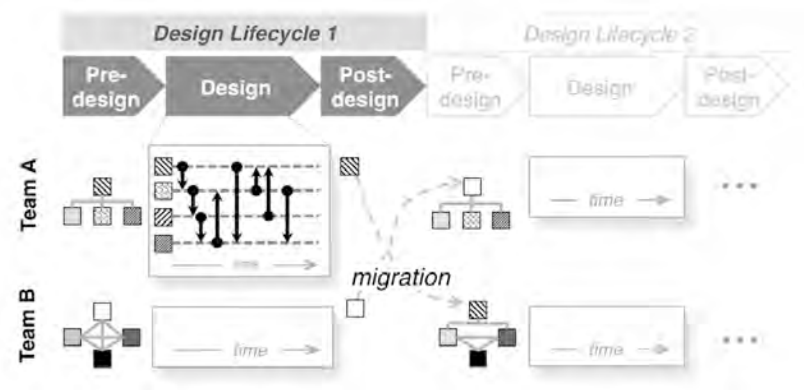

Figure 1. Design Lifecycles

3.3.1 Pre-Design Team Decisions

Collective decision-making is performed using the following iterative protocol:

1. Each Designer, randomly and in turn, proposes a system architecture, design process, and role assignment.
2. All Designers evaluate the proposal using their conceptual model of design (see Section 3.4)
3. All Designers respond with either *Approve*, *Reject*, or *Abstain* (i.e. indifferent).
4. A proposal is accepted if there is at least one *Approve* and no *Rejects*.
5. Repeat until a proposal is approved or a time limit is reached (10 cycles). If the time limit is reached, then a random choice is made for all alternatives.

3.3.2 Design Process Decision

The design process has three parameters that govern the evolution process (see Section 3.3.5):

1. *E* = {*evolve*, *do_not_evolve*}
2. *e_start* = when to start the evolution process; discrete choices = {0, 20, 50}
3. *e_period* = how frequently to perform the evolution process; discrete choices = {1, 5, 20}

3.3.3 Role Assignment Decision

There is a single parameter for role assignments: $R = \{random, systematic\}$. Designers can either be assigned decisions randomly, or can be assigned systematically, one to a subsystem. The advantage of assigning randomly is speed – the Team can start designing right away. If the Team chooses systematic assignment, it has the advantage that each Designer will have complete control over the interdependent decisions in that subsystem and thus be better able to search for the best decision combination. This comes at a cost of time at the start of the project, where Designers have to spend s = 20 periods to do the role assignment and are not able to start design work.

3.3.4 Design Phase: Design Work

After Pre-design is complete, Teams perform design work until the project time limit is reached (set experimentally). This is implemented using a population of T = 100 "Team Threads", each consisting of five "Design Threads". (The term "thread" comes from computer engineering, meaning a block of code that executes independently and potentially in parallel with other threads.) Each Team Thread has a randomly initialized design decision vector. We can think of these as 100 independent lines of investigation and analysis in the design space. Each time step in the simulation, each of the five Design Threads performs one step in a search process over possible design decisions. Each Design Thread decides to keep the proposed decision if it (locally) improves performance. Because of interdependencies, this process does not necessarily result in monotonic increase in performance. Indeed, the more dependencies and the more designers involved in those dependencies, the more likely it is that subsystem and total system performance will go through periods of decline.

3.3.5 Evolution Process

If a given Team has chosen E = *evolve*, then the population of Team Threads is modified by an evolution process, starting after *steps* = *e_start*, every *e_period* steps. Evolution is performed by probabilistic replication, where probability of being selected for replication is proportional to current system performance. The size of the population stays fixed and no new variants are introduced. Thus, 100 draws are made from a roulette-style mechanism. This has the effect of increasing search activity in the neighbourhood of currently attractive decision sets. However, given the size and complexity of the Performance Landscape, there is a risk associated with choosing E = *evolve*, and also starting too soon or doing it too frequently, namely that the Team will not allow sufficient time to search for and find truly attractive decision sets. Conversely, if a Team choses E = *do_not_evolve*, there is a risk that its design efforts will be too diversified and not concentrated enough to find the best decision sets. Designers can only learn about these trade-offs through experience, and since their experiences may differ, they can arrive at different conclusions and beliefs.

3.3.6 Post-Design Phase

After a design project is completed, the total system performance is computed and each Designer records evidence from their experience, including an evaluation of whether the project was a "Success" or "Failure". For parsimony, "Success" is defined as exceeding the average of the previous lifecycle results for all Teams, using a Cobb-Douglas function for team performance U that combines design performance p and project completion time t (i.e. number of steps to reach final/steady state system performance). Conversely, "Failure" is performance that is below the previous average team performance. The parameters α and β are chosen to give equal weight to design performance and time to completion. C is a scaling constant chosen to set expected U = 100.

$$U = C \times p^{\alpha} \times t^{\beta} \quad (1)$$

The primary experimental parameter (see Section 5) is $M = \{migrate, do\_not\_migrate\}$. When M is set to *migrate*, teams are selected for migration ($\Pr(m) = 0.2$) or to stay together ($\Pr(m') = 0.8$) to form a new Team and repeat the design project lifecycle. If a Team is selected for migration, then another Team is selected randomly to swap one team member, both randomly selected. Over several lifecycles, this migration process results in teams with heterogeneous experiences and beliefs. If M is set to *do_not_migrate*, then there is no mixing and thus Teams will consist of homogenous Designers.

3.4 Designer Agents

Designers have three sets of capabilities. First, they have the capability to perform design work as described in the previous subsection. During the Design Phase of a project, each Design agent will have one “Design Thread” per “Team Thread”, or 100 in total. These represent individual “lines of investigation” in the design space assigned to that Designer.

The second capability is to adapt a “mental model” of the design process based on their individual experience. “Mental models” are implemented as explanatory networks using the ECHO system (Thagard 1989; 2000). These models are networks of explanatory relationships between hypotheses and evidence, modulated by the “strength of activation”. A partial example is given in Figure 2, showing an evaluation of two proposals for system architecture.

The structure of the ECHO models is fixed and set by experimenters, and therefore what each Designer agent adjusts are the activation strengths. In Section 5, we discuss model extensions where the structure of mental models is determined endogenously.

A key feature of these mental models is that they permit each Designer to evaluate hypothetical situations that may not yet have been experienced or that go against what has been experienced (“counterfactual”). While accumulating more experience, especially diverse experience, will result in more refined mental models, we cannot say that mental models converge toward a true and accurate model of the Designer’s world and choice space. Instead, it is better to view a given mental model as that Designer’s beliefs about the choice space, given experience and given the limitations of the mental modeling system.

3.5 Performance Landscape of Design Decisions and Dependencies

Design decisions are represented abstractly is a fixed-size vector of N binary choices, where $N = 50$. Designs are divided in to five equal sized subsystems, so that each subsystem consists of 10 interdependent decisions. Before they start design work, Teams collectively decide on the product architecture, which has four possibilities, shown in Figure 3.

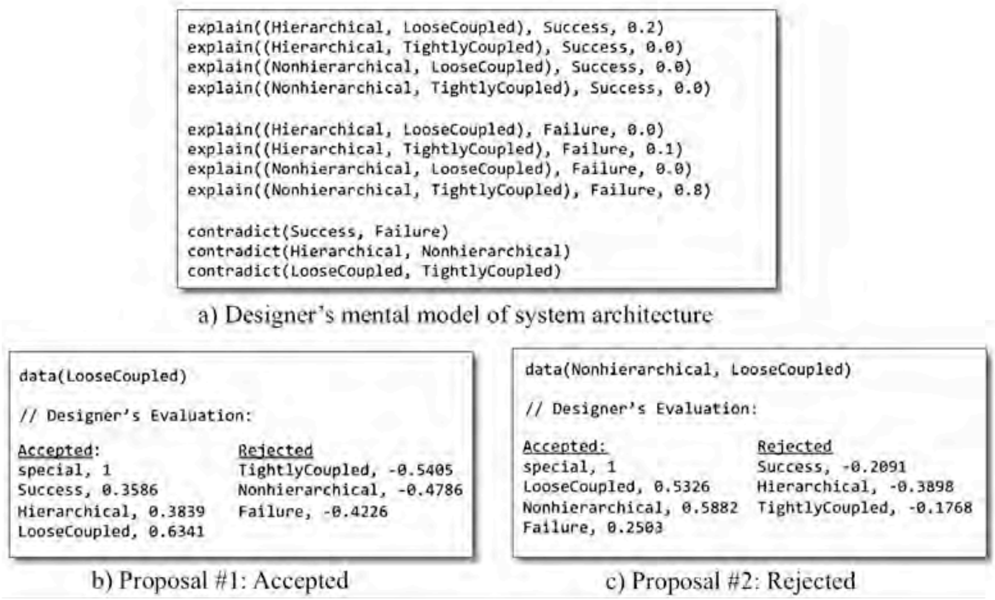

Figure 2. a) An example of Designer’s mental model for product architecture decisions, evaluating two proposals. In b) Proposal #1 is accepted because Designer assumes “Hierarchical”. The modified proposal in c) specifies “Nonhierarchical” and is rejected.

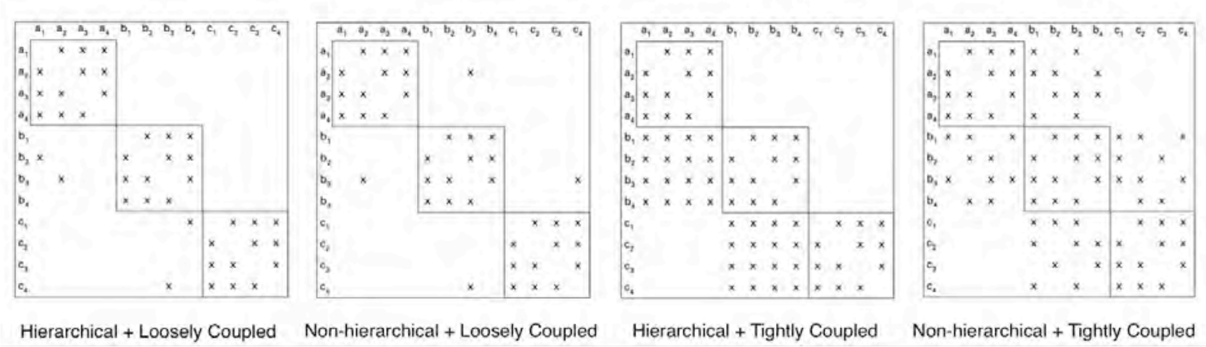

Figure 3. System architectures, displayed as design dependency matrices (Ethiraj and Levinthal 2004).

To encode the full size of the Performance Landscape, we concatenate the dependencies to each design decision. For example, if the decision vector was of length N = 4 and each decision had two dependencies, then the encoded binary string would be 4 × 3 = 12 bits long. The landscape is generated at initialization time by generating a random performance value, Uniform(0,1), for each possible encoded binary string. However, only a fixed proportion of design decisions are "viable". In all experiments, the viability proportion is 0.2. Non-viable alternatives have their performance derated by 99%. This feature is useful to reward explicit search over alternatives and penalize Designers who are ignorant of design dependencies.

The performance of the system as a whole is the average of the performance of the individual subsystems, which are in turn the average of the performance values of the individual decisions in that subsystem. Alternative performance functions are discussed in the final section of the paper.

4 RESULTS

This section describes computational experiments and results of those experiments. Prior to running these experiments, the simulation model was run many times to explore the effects of various parameters (e.g. relative performance of alternative system architectures, effects of evolution, and changing the number of Designers per Team and number of Teams.). These explorations were used to set the simulation parameters that were not varied in the experiments, especially viability proportion v = 0.2, length of design project *max_steps* = 100, and discovery rate for dependencies d = 0.002.

4.1 Experiment Design

Two treatments were evaluated using controlled, randomized simulation realizations – the treatment group *Migration*; and the control group *No Migration*.

Each trial (i.e. run) consists of 6 design lifecycles, with each lifecycle consisting of 100 steps of design work.

The number of trials for each treatment was 100. Both treatments used the same set of random number seeds. The implementation used a single set of random number generators for each run, and therefore using a given random number seed produces repeatable results and also creates identical initial conditions for each treatment. For the primary dependent variable U (equation 1), this approach allows two types of statistical hypothesis testing:

- Student t-test to evaluate differences in mean results between the two treatments
- Paired t-test to evaluate mean differences in pairs of trials that have the same random seed

We are also interested in understanding whether the pattern of Team decisions differs between the treatment groups. In other words, do Teams that migrate make different Team-level decisions compared to those who don't? To answer this question we look for significant differences in frequency for various Team-level decisions and also differences over the course of trials.

4.2 Results: Total Performance *U*

Figure 4 shows results of Student t-test to evaluate differences in mean performance in trials for the treatment (*Migration*) vs. control group (*No Migration*) for the dependent variable U, summed over 6 lifecycles in each trial. These results show that the null hypothesis (no difference in trial means) cannot be rejected. This result indicates that in general there is no performance advantage provided by team diversity, where "in general" means across many possible Performance Landscapes.

Figure 5 shows results of Paired *t*-test to evaluate mean differences from paired trials between the treatment and control groups for the dependent variable *U*, summed over 6 lifecycles in each trial. In contrast to the previous test, the Paired *t*-test, Figure 6, does reject the null hypothesis of no mean difference.

This result indicates that there are circumstances where team member migration has advantages, at least when measured by total team performance *U*. To understand these circumstances we must examine the frequency and pattern of team-level decisions.

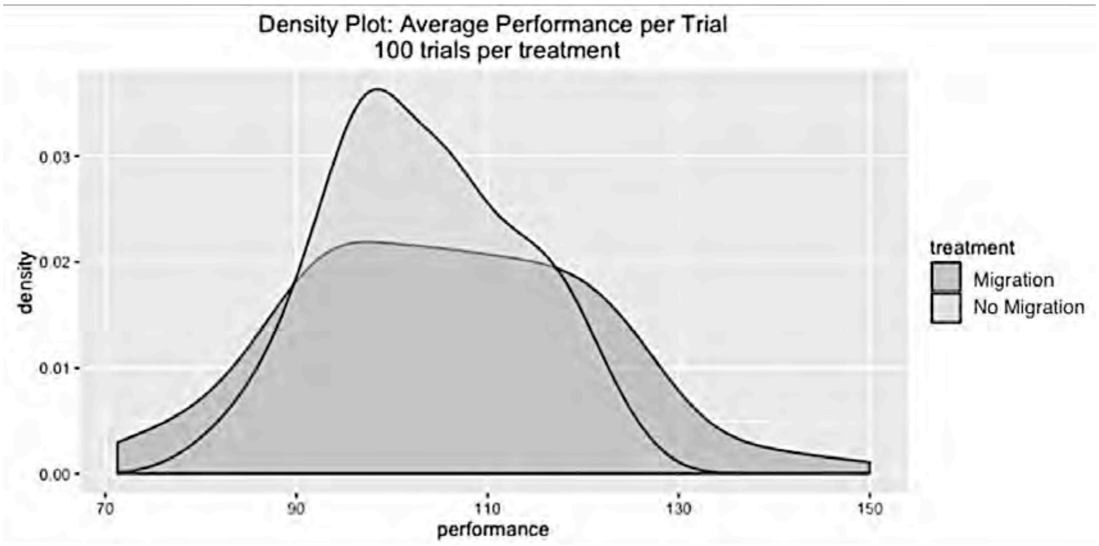

Figure 4. Density plot of average performance (U) per trial for treatment vs. control.
(Welch Two Sample t-test: t = 0.885, df = 182.3, p-value = 0.377)

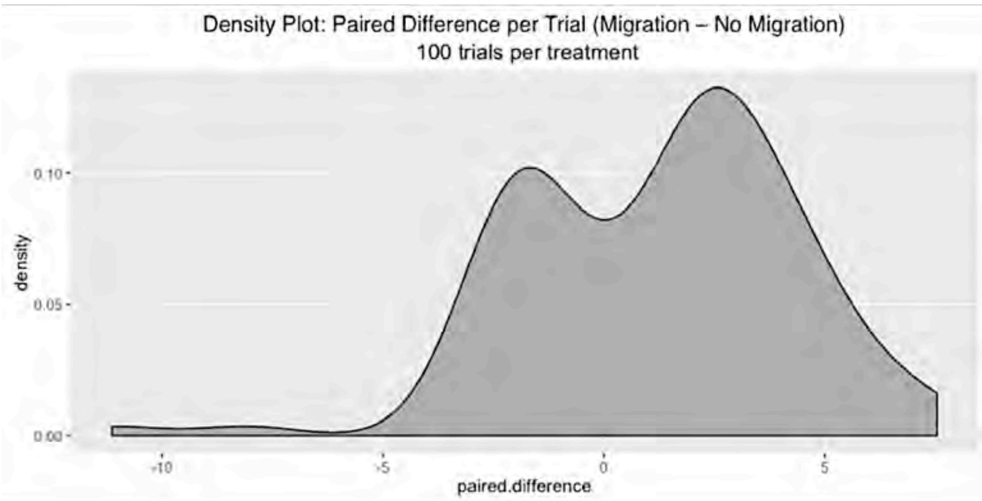

Figure 5. Density plot of paired differences: Migration – No-migration
(Paired t-test: t =3.828, df =99, p-value = 0.0002)

4.3 Results: Team-level Decision Frequency and Patterns

Figure 6 shows the frequency of team-level decision sets for *Migration* vs. *No Migration*. Referring to Section 3.3 above, a team-level decision set *D* consists of a tuple, where *A* is the System Architecture in the form of a Design Dependency Matrix, $E = \{evolve, do\_not\_evolve\}$, and *R* is role assignments: $\{random, systematic\}$:

$$D = \{A, E, e\_start, e\_end, R\} \tag{2}$$

There are 135 possible members in the decision set *D*, and thus it is infeasible for any given team to explore the full space of possibilities in any single run.
Examining these results we find that that the treatment group (*Migration*) use a wider range of team-level decisions compared to the control group (*No Migration*). Most of the team-level decisions in the *No Migration* group are concentrated in a few alternatives. Furthermore, when we inspected time-series data for individual teams and runs, we saw evidence that teams in the *No Migration* group settled into team-level decisions that did not change, even when they occasionally did not perform well. In contrast, team-level decisions in the *Migration* group changed more frequently, even when this lead to periods of under-performance.

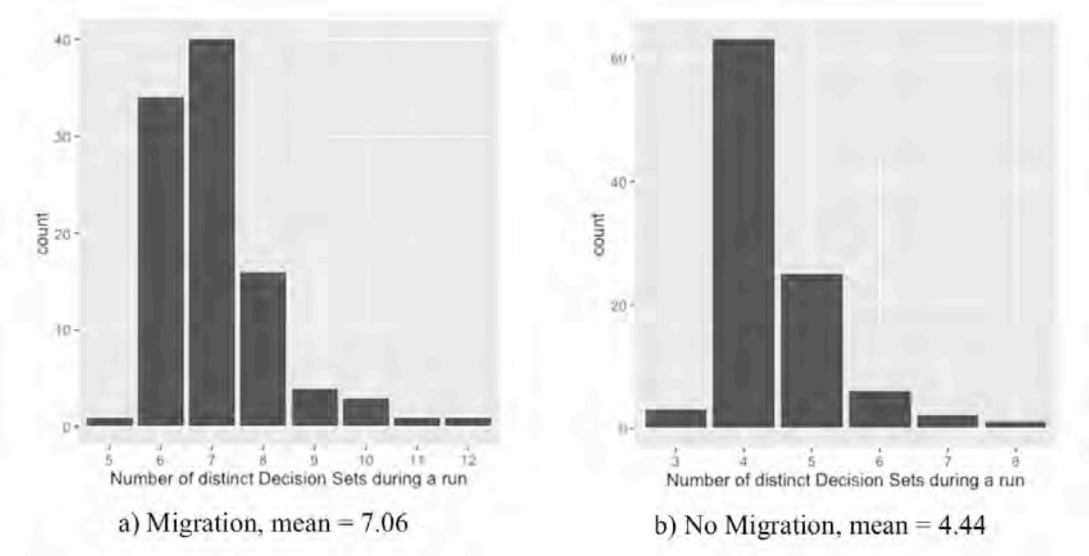

Figure 6. Frequency of the number of distinct decision sets D for each team in each run. 100 runs in each treatment.

These dynamics help explain the results in Section 4.2. Both the *Migration* and *No Migration* groups were subject to periods of under-performance but for different reasons. Teams in the *No Migration* group tended to stick with the same team-level decisions even when they under-performed, while Teams in *Migration* group tended to explore alternative team-level decisions at the expense of under-performance. This results in approximately the same mean total performance *U* over all trials, but a significant difference in paired trials.

5 DISCUSSION AND FUTURE WORK

We have used a computational model of team design with migration to examine a challenging research question regarding the effects of diversity on team decision-making and performance over several product lifecycles. Though our computational model is abstract and idealized in several respects, it has allowed us to test hypotheses that would not be feasible with human subjects in a laboratory or through field case studies.

5.1 Findings and Significance

The results in Section 4 provide qualified support for the hypothesis that heterogeneity of team member background, experience, and beliefs increases a team's ability to explore and experiment at a team-level, specifically regarding system architecture, design process, and role assignments. However this is not an unmitigated advantage because of the exploration/exploitation trade-offs that are inherent in any complex setting. The most favourable settings for team diversity are those that feature an unknown/unexplored or highly dynamic design space because diverse teams explore a wider range of team-level decisions. It is in these settings that organizations should promote migration of designers between teams to achieve that diversity.

5.2 Extensions to the Current Model

As mentioned in the sections above, the current model has many idealizations and simplifications that can be modified or extended. Here we discuss a few that have been tested or considered.

One extension would be to have alternatives and diversity in performance functions that relate individual decisions to subsystem and system performance. The current model uses average performance, and this is consistent with the model of Ethiraj and Levinthal (2004). A richer and more realistic model would have a variety of functions such as *min* ("weakest link"), *max* ("best effort"), *threshold,* and even multiplicative or nonlinear functions.

Another extension would be to add sophistication to the Designer's mental model. The current model uses only a small set of features and capabilities of the ECHO system. For example, ECHO has the ability to incorporate "analogous" predicates for analogical reasoning, which could be useful in modeling a framing discourse among Designers in the Pre-design phase. We could also add predicates related to social relationships and reputations among Designers.

While the abstract Performance Landscape used in this paper has advantages, it also has disadvantages. We as experimenters must explicitly define the performance of each decision set and also the performance function to aggregate decisions to subsystem and system-level performance. A significant change to this model would be to replace the Performance Landscape with a virtual world of Artefacts and design decisions that produce those Artefacts, along with Consumers who evaluate Artefacts according to their values and preferences. This would allow us the study the effects of team diversity and team decision-making in a more natural setting.

REFERENCES

Akgün, A. E., and Lynn, G. S. (2002), Antecedents and consequences of team stability on new product development performance. *Journal of Engineering and Technology Management,* Vol. 19 No. 3–4, pp. 263–286.

Badke-Schaub, P.; Neumann, A.; Lauche, K.; and Mohammed, S. (2007), Mental models in design teams: a valid approach to performance in design collaboration? *CoDesign*, Vol. 3 No. 1, pp. 5–20.

Bhamra, T.; Lilley, D.; and Tang, T. (2011), Design for sustainable behaviour: Using products to change consumer behaviour. *The Design Journal*, Vol. 14 No. 4, pp. 427–445.

Chidambaram, L., and Bostrom, R. (1996), Group development (I): A review and synthesis of development models. *Group Decision and Negotiation,* Vol. 6 No. 2, pp. 159–187.

Cross, N. (2004), Design studies special issue: Expertise in design. *Design Studies*, Vol. 25 No. 5, pp. 425–426.

Dong, A.; Kleinsmann, M. S.; and Deken, F. (2013), Investigating design cognition in the construction and enactment of team mental models. *Design Studies*, Vol. 34 No. 1, pp. 1–33.

Dreu, C. K. W. D. (2002), Team innovation and team effectiveness: The importance of minority dissent and reflexivity. *European Journal of Work and Organizational Psychology,* Vol. 11 No. 3, pp. 285–298.

Hansen, T., and Levine, J. M. (2009), Newcomers as change agents: Effects of newcomers' behavioral style and teams' performance optimism. *Social Influence,* Vol. 4 No. 1, pp. 46–61.

Hong, L., & Page, S. E. (2001), Problem solving by heterogeneous agents. *Journal of Economic Theory*, Vol. 97 No. 1, pp. 123–163.

Hirst, G. (2009), Effects of membership change on open discussion and team performance: The moderating role of team tenure. *European Journal of Work and Organizational Psychology,* Vol. 18 No. 2, pp. 231–249.

Joyner, R. C. (1968), Computer simulation of concept learning by individuals in a minimum social situation. In *Proceedings of the Second Conference on Applications of Simulations*, pp. 163–169.

Lawson, B. (2004), Schemata, gambits and precedent: Some factors in design expertise. *Design Studies*, Vol. 25 No. 5, pp. 443–457.

Leenders, R. T. A. J.; Van Engelen, J. M. L.; and Kratzer, J. (2007), Systematic design methods and the creative performance of new product teams: Do they contradict or complement each other? *Journal of Product Innovation Management*, Vol. 24 No. 2, pp. 166–179.

Lockton, D.; Harrison, D.; and Stanton, N. A. (2010), The Design with intent method: A design tool for influencing user behaviour. *Applied Ergonomics*, Vol. 41 No. 3, pp. 382–392.

Metcalfe, J. S. (2007), Replicator Dynamics. In *The Elgar Companion to Neo-Schumpeterian Economics*. Edward Elgar Publishing.

McGrath, J. E. (1991), Time, interaction, and performance (TIP): A theory of groups. *Small Group Research,* Vol. 22 No. 2, pp. 147–174.

Nardi, B. A. (1996), Studying Context: A Comparison of activity theory, situated action models, and distributed cognition. In Nardi, B. A., ed., *Context and Consciousness: Activity Theory and Human-Computer Interaction*. MIT Press.

Reagans, R.; Argote, L.; and Brooks, D. (2005), Individual Experience and Experience Working Together: Predicting Learning Rates from Knowing Who Knows What and Knowing How to Work Together. *Management Science,* Vol. 51 No. 6, pp. 869–881.

Rink, F.; Kane, A. A.; Ellemers, N.; and van der Vegt, G. (2013), Team Receptivity to Newcomers: Five Decades of Evidence and Future Research Themes. *The Academy of Management Annals,* Vol. 7 No. 1, pp. 247–293.

Schank, P. and Ranney, M. (1991), The Psychological Fidelity of ECHO: Modeling an Experimental Study of Explanatory Coherence. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, pp. 892–897. Chicago, Illinois: Lawrence Erlbaum Associates.

Schippers, M. C.; Den Hartog, D. N.; and Koopman, P. L. (2007), Reflexivity in Teams: A Measure and Correlates. *Applied Psychology,* Vol. 56 No. 2, pp. 189–211.

Singh, V.; Dong, A.; and Gero, J. S. (2011), How important is team structure to team performance? In *Proceedings of the 18th International Conference on Engineering Design (ICED 11)*.

Smith, G. J., and Gero, J. S. (2005), What does an artificial design agent mean by being 'situated'? *Design Studies,* Vol. 26 No. 5, pp. 535–561.

Thagard, P. (1992), *Conceptual Revolutions*. Princeton, NJ: Princeton University Press.

Thagard, P. (2000), *Coherence in Thought and Action*. Cambridge, MA: MIT Press.

van der Haar, S.; Li, J.; Segers, M.; Jehn, K. A.; and den Bossche, P. V. (2015), Evolving team cognition: The impact of team situation models on team effectiveness. *European Journal of Work and Organizational Psychology,* Vol. 24 No. 4, pp. 596–610.

Verhulst, E., & Boks, C. (2012), The role of human factors in the adoption of sustainable design criteria in business: evidence from Belgian and Dutch case studies. *International Journal of Innovation and Sustainable Development*, Vol. 6 No. 2, pp. 146–163.

Walz, D. B.; Elam, J. J.; and Curtis, B. (1993), Inside a software design team: Knowledge acquisition, sharing, and integration. *Communications of the ACM,* Vol. 36 No. 10, pp. 62–77.

Wever, R.; Kuijk, J. van; and Boks, C. (2008), User‐centred design for sustainable behaviour. *International Journal of Sustainable Engineering*, Vol. 1 No. 1, pp. 9–20.

ACKNOWLEDGMENTS

This paper is based upon research supported by the National Science Foundation under Grant No. CMMI-1400466. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.