Journal of Mechanical Design

Capturing Human Sequence-Learning Abilities in Configuration Design Tasks through Markov Chains

Christopher McComb
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA
ccm@cmu.edu
ASME Member

Jonathan Cagan
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA
cagan@cmu.edu
ASME Fellow

Kenneth Kotovsky
Department of Psychology
Carnegie Mellon University
Pittsburgh, PA
kotovsky@cmu.edu

Abstract

Design often involves searching for a solution by iteratively modifying and adjusting a current design. Through this process, designers improve the quality of the current design as well as learning what patterns of operations are most likely to lead to the quickest future improvements. Prior work in psychology has shown that humans can be adept at learning how to apply short sequences of operations for maximum effect while solving a problem. This work explores the sequencing of operations specifically within the domain of engineering design by examining the results of two human studies in which participants created solutions to configuration design problems. First, a statistical analysis of the data from those studies uses Markov chains to show that meaningful operation sequences exist, and can be accurately described using first-order models. Second, this work uses an agent-based modeling framework in conjunction with Markov chain concepts to simulate the performance of teams with and without the ability to learn sequences. These computational studies confirm the assumption that sequence-learning abilities are helpful during design.

1 Introduction

Design often involves searching for a solution by iteratively modifying and adjusting a current design. Through this process designers search the design space, attempting to iteratively improve on their current solution. However, through the process of searching, designers also learn how to efficiently navigate the design space through the operations that they use to modify their solutions. This work studies the ability of designers to learn and apply beneficial sequences of operations, and examines the performance implications of such behavior.

The current work specifically stems from a study in which small teams of engineering students were instructed to design a truss [1]. Previous work hypothesized that the order in which operations were performed in that study may have had a large impact on the quality of solutions [2]. By focusing on the ordering of operations, the analysis in the current work is conducted at a more fine-grained resolution and at shorter timescales than other work on design stage or task sequencing (which is reviewed in the Background section of this paper). It is at this level that engineers and designers directly interact with potential solutions, so the selection and application of the best actions is particularly crucial for the creation of high quality solutions. Two questions are specifically addressed in this work:

1. *How much representational complexity is necessary to quantify the sequential patterns that designers employ during solving?* Previous work has used sequential models with varying degrees of complexity to examine designer activity [3–6]. However, no direct analysis has been conducted to assess what degree of complexity is necessary to offer an accurate aggregate representation of designer activity. The current work utilizes Markov chain concepts to verify the fundamental assumption that sequential treatments are necessary, and to uncover the necessary level of complexity.

2. *Is the employment of sequences of operations beneficial to designers?* Effective design must find a balance between exploration of a design space and exploitation of known features of the design space to achieve a solution [7–9]. Sequence-learning may serve to augment exploitation in design, similar to the role that it plays for solving puzzle problems [10,11]. However, it is also possible that this augmentation occurs at the expense of effective exploration, as designers may apply learned sequences to greedily improve solution quality rather than searching broadly for solution alternatives. Studying the performance implications of sequence-learning is

complicated by the fact that sequence-learning can take place implicitly [12,13], which makes it challenging to control, observe, and assess. This work uses a computational model of engineering design teams in conjunction with Markov concepts to accurately assess the performance implications of sequence-learning abilities.

This work makes use of the Markov chain concept as a tool for understanding and simulating the order in which humans perform operations in a design context, thus helping to address the research questions. It should be noted that Markov models do not directly extract fixed-length sequences of operations. Instead, such models implicitly represent fixed-length sequences using probabilistic chains. The mathematical underpinnings of Markov chains are described in greater detail in the Background section, along with relevant information pertaining to sequencing and design.

The main body of this paper is presented as two investigations, each of which is aligned with one of the research questions posed previously. The first explores what degree of complexity is necessary to accurately characterize the sequences used by designers. To answer that question, a statistical analysis is conducted on the human data from two cognitive studies. This analysis indicates with high confidence that participants in both studies utilized sequences of operations when creating solutions. The results also show that operation sequencing in both studies can be characterized as a first-order Markov process. Higher-order Markov models display accuracy that is statistically equivalent to first-order models. The second investigation analyzes whether or not sequences of operations are beneficial to the designer. This is accomplished by performing computational simulations of design team behavior using the Cognitively-Inspired Simulated Annealing Teams (CISAT) modeling framework, an agent-based platform that approximates the process and performance of engineering design teams [2]. The insight that human operation sequencing can be treated as a first-order Markov process is used to provide CISAT agents with the ability to learn sequences, enabling a computational comparison between teams with and without sequence-learning abilities. These simulations demonstrate that the ability to learn sequences during design was extremely beneficial to teams in the cognitive studies, and that similar computational implementations may be of use for automated design synthesis.

2 Background

The ability to learn sequences of operations is essential to human performance across a variety of both mundane and specialized tasks [14]. A related behavior is the ability of humans to collect many pieces of related information within a single unit of memory. This process is commonly referred to as *chunking*, and has been observed across a variety of domains [15–17]. Observations of chunking behavior in humans have even led to the enhancement of computational design algorithms [18]. While chunking is an important aspect of design, this work focuses on an equally important aspect: sequencing of operations. Sequencing may be thought of as temporal chunking, but the two behaviors are functionally distinct.

2.1 Sequence Learning

In some domains, the presence of sequential behavior has been shown to be indicative of expertise [19]. However, in other studies of individual problem-solving that used isomorphs of the Tower of Hanoi puzzle [10,11] or the Thurstone Letters Series Completion task [20] it was shown that participants were able to acquire and begin employing move sequences within a short period of time. In studies using the Tower of Hanoi puzzle, participants spent most of their time learning to sequence moves appropriately, and a relatively short amount of time in finally solving the puzzle [10]. When comparing easy and difficult isomorphs of that puzzle, it was discovered that participants solving hard isomorphs spent longer in the learning phase, but that both conditions took approximately the same amount of time in the final solving phase [10]. Studies using the Thurstone task identified that participants' problem-solving behavior consisted broadly of two steps. In the first step participants recognized some order in the letter series and codified it as a rule (i.e., generating a pattern), and in the second step they used that rule to extrapolate the letter series (i.e., generating a sequence) [20]. This two-step process was validated with computer simulations [20], and has since been revisited and confirmed [21]. Other work explicitly studied the order of operations used in a geometric analogy task [22]. Although the task itself placed no constraints on the order in which operations could be performed, it was found that there was a strong preference amongst participants for a specific order [22]. Further, participants made to use a non-preferred order performed with lower speed and accuracy, indicating that the preferred order was an important construct [22].

There is evidence that humans learn sequences implicitly, without the need for direct attention [12,13] but there is also evidence that explicitly learning sequences can boost effectiveness [23,24]. This evidential duality gives credence to the theory that sequence learning can occur through a variety of cognitive pathways [14,25].

2.2 Sequencing in Design

Research on sequencing in design has examined the ordering of design stages, the ordering of specific tasks, and the sequencing of discrete design operations. These three types of design sequences can be conceptualized along a continuum that describes the degree of abstraction, from design stages (the most abstract and generalized of the three) to design operations (the least abstract and most detail-specific). These types of sequences can also be differentiated in terms of the timescale at which the sequenced object is enacted, with design stages at the longest timescales and design operations at the shortest.

The sequencing of design stages is commonly examined through design studies with either teams or individuals. In a study that coded intra-team design communication as either content-focused or process-focused, it was noted that teams had a high probability of remaining on one focus for several communicative acts before switching to the other focus [26]. The same study also coded communication in terms of steps in the design process, and noted a similar pattern of repeated communicative acts within a step [26]. Another study tasked participants with the design of a playground, and activity was coded with respect to design stages [27]. Expert sequences flowed smoothly from one stage to the next, while novice sequences were more choppy and unstable [27]. In a comparison of undergraduate and PhD students, it was noted that there was significant variability in the sequence in which design stages were visited [28]. Interestingly, none of the participants followed a linear design process [28]. However, a similar study showed that participants who transitioned linearly through the design process produced more effective solutions [29].

The sequencing of design tasks however takes place at shorter timescales than the sequencing of design stages, with numerous design tasks usually performed within a single stage. Intelligent ordering of tasks can increase the concurrency with which tasks can be completed [30], minimize the time and cost involved in developing a product [31], and increase the

availability of information for important decisions [32]. Waldron and Waldron observed the sequence of tasks involved in the design of a complex walking vehicle [33]. They noted that there was not a distinct separation between conceptual and detailed design and that tasks may span across design stages [33]. Theoretical work on task sequencing has shown that optimal strategies for ordering tasks may be tied to problem complexity [32]. Other work has utilized genetic algorithms to optimize task sequences for a variety of different objectives [34].

The sequencing of discrete design operations takes place on the shortest timescales and has the most intimate impact on potential solutions. It is at this level that engineers and designers directly interact with solution attributes, so selection and application of the best operations is particularly crucial for the creation of high-quality solutions. Much of the work that has examined the sequencing of design operations makes use of some type of protocol encoding scheme in order to render the resulting sequences meaningful. Function-Behavior-Structure (FBS) concepts [35,36] are commonly used to create coding schemes to study sequencing at this scale. The FBS design ontology specifically describes design as a process with the ultimate goal of transforming a set of design requirements into a design description [35]. The description cannot proceed directly from the set of requirements, but instead arises as a result of considering a number of issues associated with the design requirements – the required functionality, the expected behavior, the observed behavior, and the structure of the designed object. The transitions between these issues are referred to as processes. First-order sequential behavior in the FBS ontology (the transitions between issues) has been modeled via Markov chains [3–5]. Second-order sequential behavior (the probability that specific processes will precede specific issues) has also been investigated [37]. Simulations have also considered the effects of memory on sequencing behavior in computational agents using higher-order Markov chains [6]. Aside from studies of human designers, the extraction and implementation of beneficial rule pairs has been explored with respect to design automation [38].

This work will also use Markov concepts to study the ordering of operations during design tasks. Instead of using a coding scheme that requires human assessment, we code design operations according to their quantifiable effect on the form of the current design solution. Markov chains concepts are used to study the sequencing of operations for design tasks, and also

to implement sequence-learning abilities within CISAT, a computational model of design teams [2].

2.3 Markov Processes

A Markov process is a stochastic process in which a system transitions between a finite number of discrete states. Markov processes are commonly modeled using Markov chains [39]. In a first-order Markov chain model, the probability of transitioning to a future state depends only on the current state of the system, and not on previous states [39]. The probability of transition to a future state from a current state is given by the transition matrix, $\boldsymbol{T}$, where the value of T_{ij} is the probability of transitioning from state i to state j. Markov chains were first introduced more than a century ago [40], and have since been used in a number of applications including computer performance evaluation [41], web search [42], chemical processes [43], and design team communications [44,45].

Figure 1 depicts a first-order Markov chain with three states (S_1, S_2, and S_3). The entries of the transition matrix are shown next to arrows that indicate the relevant transition. Self-transitions are allowed in the model, meaning that the system remains in the same state across multiple time steps. In this work, Markov chains are used to describe the order in which modifying operations are applied to designs. Operations constitute the states of the Markov chain model, and the transition probabilities in $\boldsymbol{T}$ describe the probability that one operation will be followed by another.

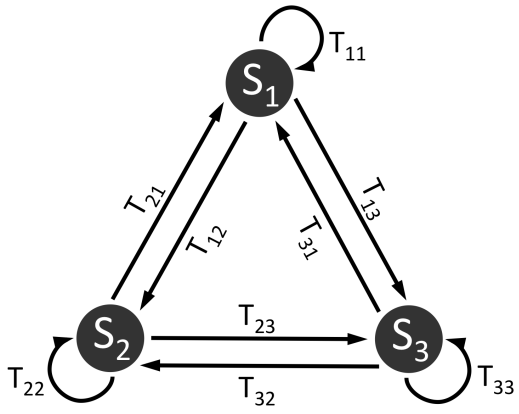

FIGURE 1. EXAMPLE MARKOV CHAIN WITH 3 STATES.

Higher-order Markov chain models assume that the selection of the next state is dependent on the current state as well as some number of past states, thus encoding some degree of "memory" in the model [46]. Within the domain of design, the inherent memory of higher-order models could provide a useful analog for a portion of the expertise and memory of human designers. Higher-order Markov chain models are used in this work to characterize how much inherent memory is necessary to describe the order in which designers apply modifying operations to a solution concept. Zero-order Markov chains are also used in this work. These models do not encode a sequential representation of data, but instead encode the non-conditional frequency with which operations are applied (much like the probabilities associated with each side of a weighted die).

3 Data Sets

The operation data sets analyzed in this work were derived from two previously-conducted cognitive studies. The first study tasked engineering students with designing a truss, and was originally created to examine design in the face of dynamic problems [1]. The second study tasked a different group of engineering students with the design of an internet-connected home cooling system, and was originally designed to assess team coordination and communication [47]. A brief review of both studies is given here. Neither study was designed explicitly for the analyses applied in this paper – rather, the current work mines patterns of human behavior from those pre-existing data sets. In addition, the differences between the two studies are an advantage to the current work because the respective data provides a broad basis from which to draw more general conclusions.

While the data was collected from *team*-based experiments, the focus of the current work is on *individual* sequence-learning. This type of individual-level analysis can be performed on the team-based data for two reasons. First, a separate series of operations was logged independently for every participant, rather than aggregate collection of data at the team level. Second, the fraction of time spent working individually was much larger than the fraction of time spent interacting with teammates for both studies, so learned sequences were largely the result of individual activity and effort.

3.1 Truss Design Study

This study tasked 16 teams of 3 mechanical engineering students with the design of a truss structure. Design was conducted over the course of six, 4-minute design sessions. During the study, the design problem was changed twice without warning through the introduction of new problem statements. New problem statements were introduced in this manner to study problem-solving and design in response to a dynamic design task [1]. The initial problem statement simply asked teams to design a truss with two loading points and three supports. The first change presented participants with the same layout, but also instructed them to account for the fact that any one of the supports could be removed at any time. The second change modified the problem so that teams had to build their truss around an obstacle. Teams were given a target mass and factor-of-safety for the initial problem statements and for each of the subsequent modified problem statements. Over the course of the study, participants were permitted to interact freely with members of their team [1]. Estimates of communication frequency made in previous work vary from one interaction for every 30 individual actions, to once for every 100 actions, depending on the team [2]. Because the problem statement changed during design, it is expected that this data set will yield sequencing information that applies generally to a variety of truss design problems.

To facilitate design, every participant was given access to a computer-implemented truss design program that was created for the purpose of the study. This program allowed the participants to build, assess, and share truss designs within their teams. In addition to facilitating design, the truss design program was used to continuously record the operations that participants chose and applied in order to modify their designs, thus allowing a full account of design operations to be accumulated. The operations available to participants were:

1. Adding a joint
2. Removing a joint
3. Adding a member
4. Removing a member
5. Changing the size of a single member
6. Changing the size of all members
7. Moving a joint

This information was post-processed to extract a string of move operators (denoted by the integers 1 through 7) for each of the 48 participants in the study. A typical solution sequence consists of 400 to 500 operations. A short example operation sequence is provided in Figure 2.

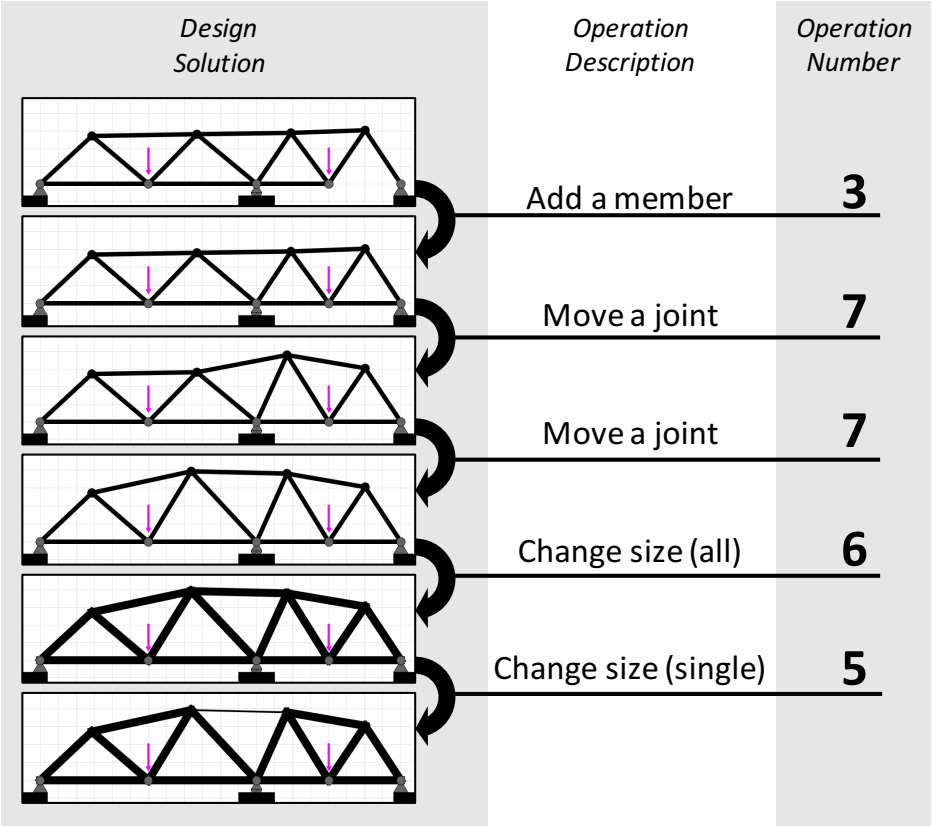

FIGURE 2. EXAMPLE TRUSS OPERATION SEQUENCES, WITH NUMBERS CORRESPONDING TO THE LIST OF TRUSS OPERATIONS.

3.2 Home Cooling System Design Study

This study tasked 54 mechanical engineering students (either individually or in teams) with designing a system of connected products to maintain the temperature within a residential structure. Participants were allowed to use and connect three distinct product types to create their solutions: sensors (which sensed the temperature of the room in which they were placed), coolers (which cooled rooms in the home), and processors (which made decisions about which coolers to activate based on information from sensors). Design was conducted over the course of a 30-minute session. Several experimental conditions were established to control the frequency with

which participants interacted with their teammates (from zero interaction to interacting once for every 5 individual actions). To ensure a common basis for comparison between conditions every participant was allowed to perform only 50 design operations.

Every participant was given access to a computer-implemented design program in this study that allowed them to build, assess, and share solutions. It was also used to continuously record the operations that participants used. The operations available to participants were:

1. Add processor
2. Add sensor
3. Add cooler
4. Remove processor
5. Remove sensor
6. Remove cooler
7. Move sensor
8. Move cooler
9. Tune cooler

This information was post-processed to extract a string of move operators (denoted by the integers 1 through 9) for each of the 54 participants in the study. Every solution sequence consisted of exactly 50 operations. A short example sequence is provided in Figure 3. The solution diagrams in the left column depict a plan view of the structure, with shading indicating the relative temperature in different rooms.

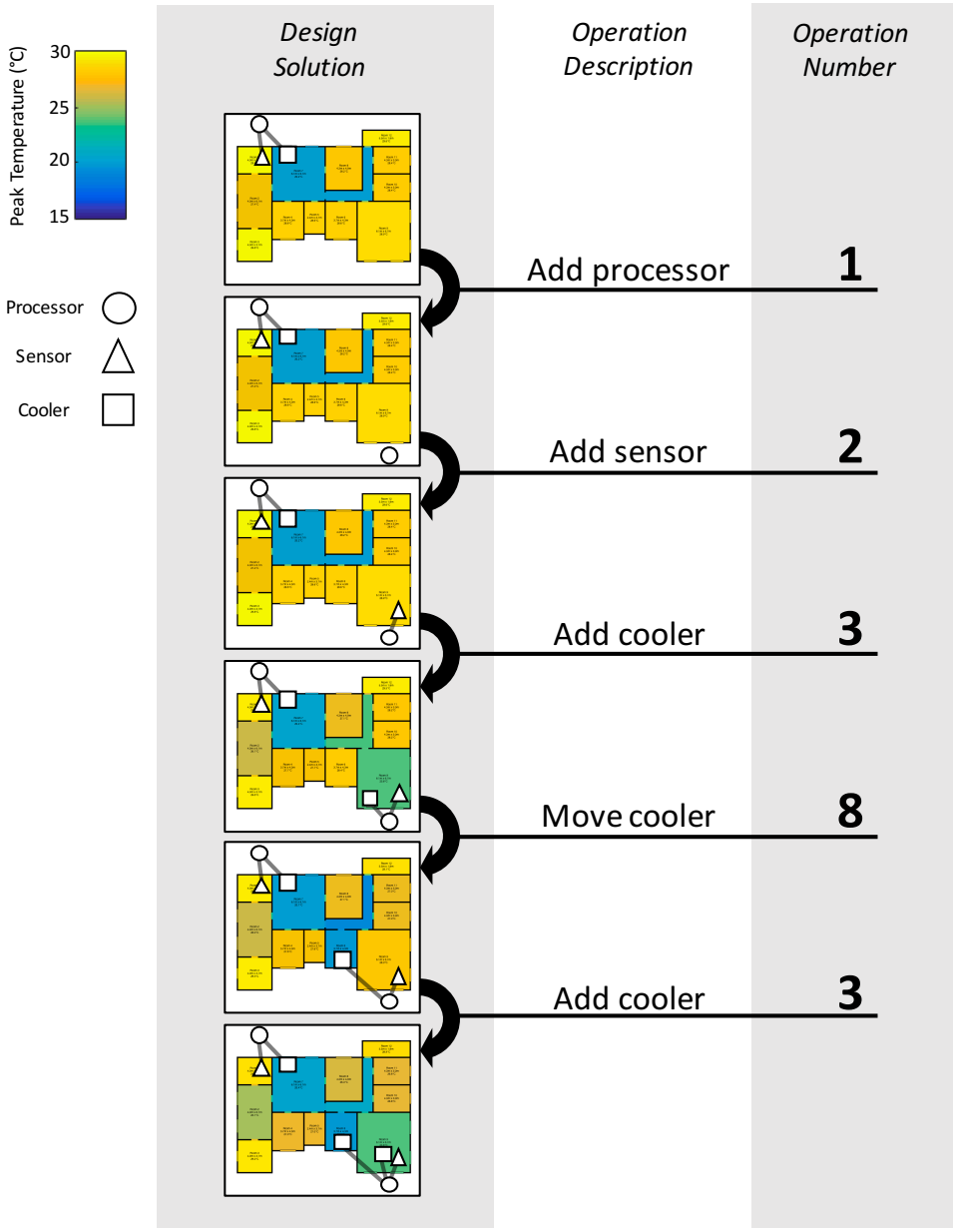

FIGURE 3. EXAMPLE COOLING SYSTEM OPERATION SEQUENCE, WITH NUMBERS CORRESPONDING TO THE LIST OF SYSTEM OPERATIONS AND SHADING INDICATING ROOM TEMPERATURES.

4 Investigation 1: Representation of Operation Sequences

This paper first analyzes human operation data from the two design studies with the objective of determining what degree of representational complexity is necessary to accurately model the sequences employed by designers.

4.1 Methodology

Markov chain models were trained on the human data in order to provide a statistical representation of the sequence in which operations occurred in the cognitive study. The following discussion of the training process is based on material in [39], but is presented in terms of design operations (rather than Markov process states) to aid understanding of its relevance to the current work. The procedure specifically outlines the training of first-order Markov models, but it can also be applied to higher-order models with small modifications.

A Markov chain is defined by the values of its transition matrix, $\boldsymbol{T}$. Element T_{ij} in the matrix defines the probability that the next operation will be operation j, given that the previous operation was operation i. Therefore, the elements of the transition matrix for a Markov chain can be estimated based on observed data using

$$T_{ij} = \frac{N_{ij}}{N_i}, \tag{1}$$

where N_{ij} is the number of times that operation j is observed to follow operation i, and N_i is the number of times that operation i is observed. The diagonal of the transition matrix contains probabilities for cases where $i = j$, indicating that an operation is followed by itself.

The log-likelihood is a quantity that indicates the probability that a model could have produced a given set of data, and can thus be used to compare different models. This is essentially a measure of the accuracy with which the model can reproduce a given data set. The log-likelihood for a Markov chain model ($\mathcal{L}_{MC}$) is

$$\mathcal{L}_{MC} = \sum_{i=1}^{M} \sum_{j=1}^{M} N_{ij} \cdot \ln\left(T_{ij}\right), \tag{2}$$

where N_{ij} and T_{ij} are defined earlier and M is the number of different operations.

The procedure for training higher-order Markov chains follows essentially the same pattern as that for first-order Markov chains. The key difference is that the states of the model are no longer single design operations, but n-tuples of operations, where n is the order of the Markov chain. Training of a zero-order Markov chain model simply consists of estimating the frequency with which each operation occurs, without any assumption of conditional dependence on earlier operations in the sequence.

Markov models were trained on both data sets, from zero order (i.e. a model assuming that future operations have no dependence on past operations) to fourth order (i.e. a model assuming that future operations dependent on the last four operations). Models were trained using leave-one-out cross-validation [48]. For a data set consisting of n samples, this cross-validation approach trains a model with $n - 1$ samples, and then tests the model on the sample that was not used for training. This procedure is repeated until every sample has been used for testing, providing n evaluations of the testing accuracy, for which the mean and standard error can be computed. It should be noted that leave-one-out cross-validation is a special case of k-folds cross validation [49] for which k is equal to the number of samples (n). Using $k = n$ provides an accuracy estimate with lower bias and a more conservative variance than values of $k < n$ [50].

In this work, each sample is composed of the data from one study participant (consisting of many operations). Thus, the validation approach used here estimates how accurate the model might be for describing the behavior of a previously-unseen individual. It should be noted that during training (and for communicating final results) the transition probabilities are computed using the data from multiple study participants.

4.2 Results for Truss Design Study

A plot of log-likelihood for models of increasing order is shown in Figure 4(a) with error bars indicating standard error. The dashed line shows the log-likelihood of the model on the training data set, and the solid line shows the log-likelihood on the testing data set. Significant differences between adjacent models are shown with dotted brackets. Models with higher testing log likelihood provide a better fit to unseen data, and should be preferred.

Figure 4(a) shows a steep increase in log-likelihood from the zero-order Markov model (which by nature cannot model any sequential behavior) to the first-order Markov model (which provides the simplest representation of sequencing behavior). This indicates that strong sequencing patterns do exist in the data from the truss study. However, after first-order the testing log-likelihood plateaus, while the training log-likelihood continues to increase. This indicates that overfitting occurs in higher-order models. In other words, higher-order models begin to fit attributes of the training data that are not general, and thus exhibit lower accuracy on the testing data set. There is a slight increase in the mean testing accuracy between the first-order and second-order models, but this increase is not significant ($F = 0.31$, $p > 0.5$).

As noted previously, the testing log-likelihood in Figure 4(a) plateaus after the first-order model, with no further significant differences apparent in the testing log-likelihood curve. Therefore, the first-order model is preferred, as it provides a degree of accuracy that is statistically equivalent to higher order models, but it does so with much less complexity. Designer activity in the truss design task can therefore be accurately modeled as a first-order Markov process.

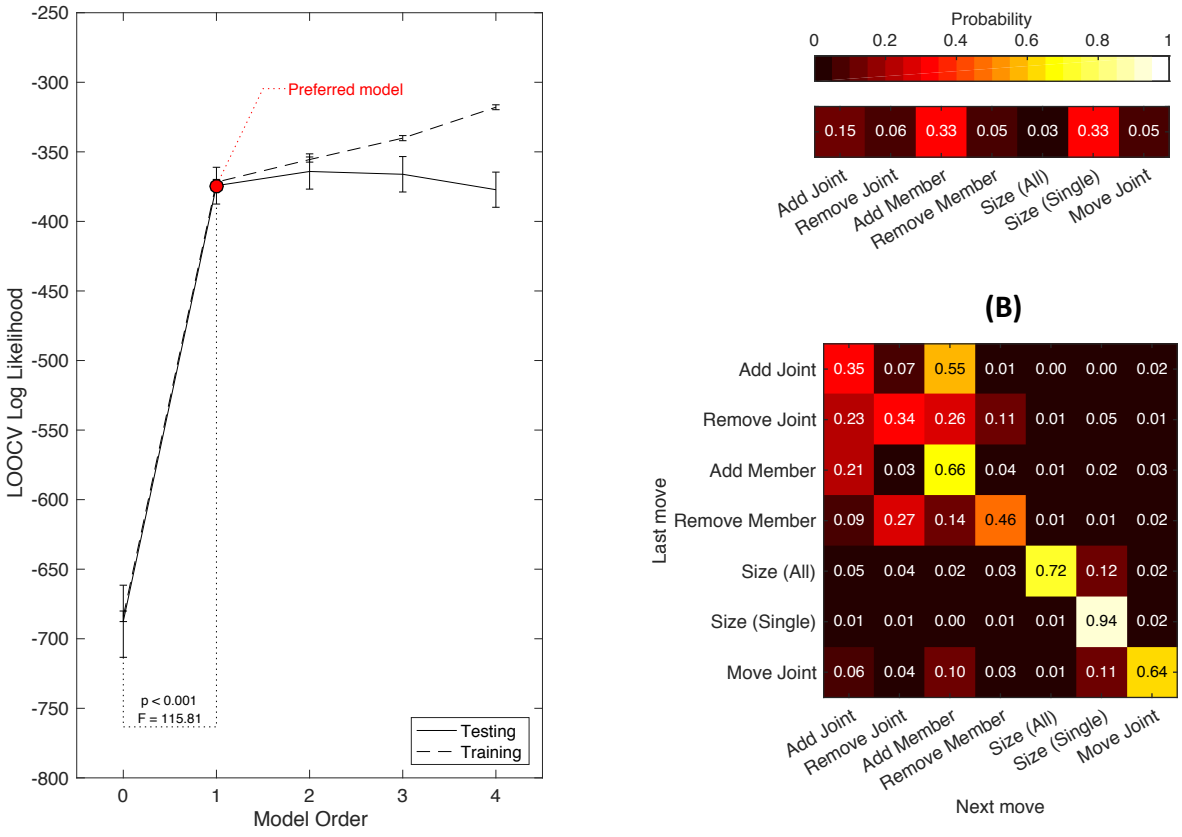

FIGURE 4. SUMMARY OF RESULTS FOR TRUSS DESIGN STUDY, INCLUDING (A) LOG-LIKELIHOOD COMPUTED FOR MODELS OF INCREASING ACCURY, (B) STATE FREQUENCIES FOR ZERO-ORDER MARKOV MODEL, AND (C) TRANSITION MATRIX OF FIRST-ORDER MODEL.

Comparing the zero-order model (which encodes a non-sequential representation) to the first-order model (which encodes a representation of designer activity that is both parsimonious and accurate) enables an examination of why sequencing of operations was important for the truss design task. The operation frequencies associated with the zero-order Markov model are shown in Figure 4(b), and the transition matrix of the first-order Markov model is shown in Figure 4(c). The shading inside the squares indicates the magnitude of the probability, which is also noted numerically within each square.

Examining the differences between Figures 4(b) and 4(c) provides some insight as to why the first-order model is so much more veridical than the zero-order model. The most striking aspect of the transition probability matrix of the first-order model is that it is largely diagonal. This indicates that the same operation was likely to be applied several times before a different

operation was selected. This aspect of the participants' behavior simply could not be captured in the zero-order model. For instance, the zero-order model indicates a 33% chance that the next operation will be to add a member, regardless of the previous operation. However, the column of the first-order transition matrix that corresponds to adding a member shows that that operation is most likely chosen after adding a joint, removing a joint, or adding a member, and rarely selected if the last move involves changing the size of truss members. Thus, the selection of the subsequent operations is highly conditional on the last operation performed. Similarly, the zero-order model shows a 33% chance of changing the size of a single member. However, the first-order transition matrix more incisively encodes the operational data by showing that that operation is rarely selected after adding a joint, removing a joint, adding a member, or removing a member.

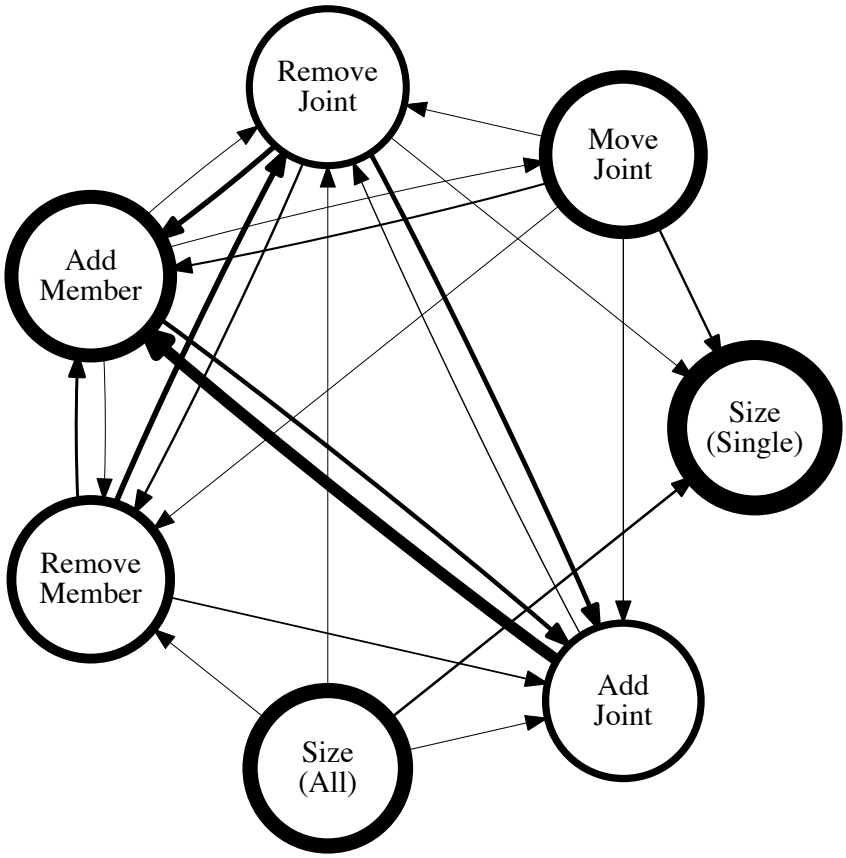

FIGURE 5. GRAPHICAL REPRESENTATION OF THE FIRST-ORDER MARKOV MODEL FOR THE TRUSS DESIGN STUDY, SHOWING THE MOST LIKELY TRANSITIONS. TRANSITION PROBABILITY IS INDICATED BY THE LINE THICKNESS OF THE CORRESPONDING ARROW (IN THE CASE OF SELF-TRANSITIONS, THE THICKNESS OF THE BORDER OF THE CORRESPONDING OPERATION).

In addition, a more detailed graphical representation of the first-order Markov chain model is provided in Figure 5. Arrows are used to indicate transitions between states and line thicknesses represent the relative likelihood of those transitions, with thicker lines indicating the more likely transitions. Only the strongest transitions were included in the representation (those with transition probabilities above the median, approximately 0.03). The probabilities pertaining to self-transitions are indicated by the thickness of the border of the corresponding operation. Figure 5 provides further insight as to the pattern of operations in the study. Actions involving the topology of the truss (adding and removing joints and members) are connected by the thickest arrows, indicating that there is a strong probability of transition between these operations. This indicates that these topology operations were usually employed together during the design study. In contrast, operations that only change the shape of the truss (changing the size of members, or moving joints) are connected by arrows that are much thinner, indicating that there is far less interaction between these operations. These operations are also much more likely to be applied multiple times in a row, as indicated by the probability of a self-transition. Finally, there is a very low probability of leaving the state corresponding to changing the size of a single member ("size (single)"). This indicates that participants were very likely to repeatedly apply this operation to fine-tune the size of the members in their designs.

Higher-order Markov models are capable of explicitly representing long sequences of operations. On the other hand, first-order Markov models assume that the selection of a subsequent operation is dependent on only the last operation, so that each operation is probabilistically linked to the next. Therefore, only pairs of operations can be represented explicitly. However, operation sequences of arbitrary length can be created by stringing together several of these probabilistically-linked pairs. Graphically, the process of constructing these sequences consists of tracing a path through the graph-based representation of the transition matrix shown in Figure 5. A set of high likelihood exemplar sequences produced through this process is provided in Figure 6. Operations are shown in rectangular boxes and the probability that the following operation would occur is given with a percentage over the linking arrow. The percentage of participants from the cognitive study who employed the sequence is also noted.

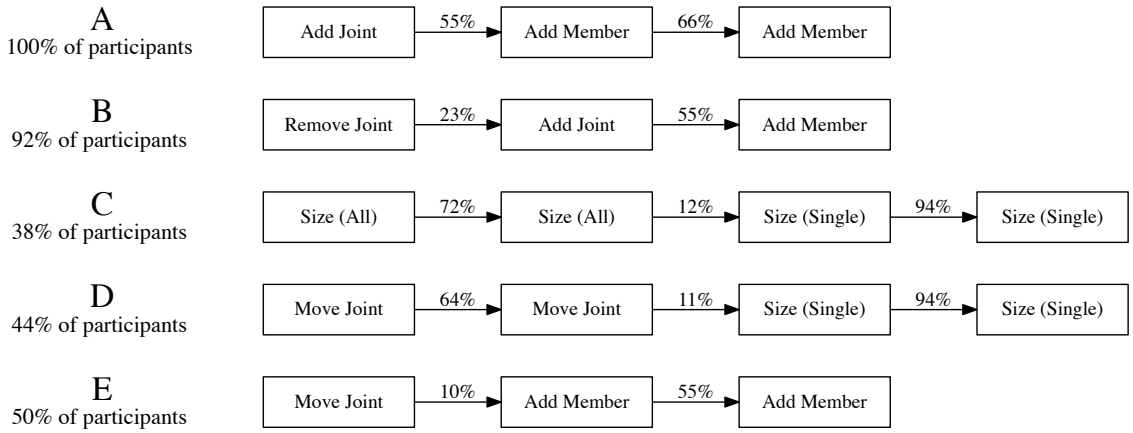

FIGURE 6. EXEMPLAR SEQUENCES EXTRACTED FROM FIRST-ORDER MARKOV MODEL.

These sequences are multi-operation patterns of action that might be expected in a truss design task. Sequence A consists of a joint addition followed by several member additions. This kind of pattern could arise as a designer constructs their truss, adding a joint and then attaching it to the existing truss with new structural members (and was employed by every participant in the cognitive study). Sequence B is similar to Sequence A in that it consists of topology operations, but instead begins with a joint removal (which also removes all attached members) following by a joint addition and a member addition. This signifies revision of a section of the truss – removing a section of the truss with poor performance and then rebuilding it in an attempt to improve performance characteristics. Sequence B was employed by 92% of the cognitive study participants. Sequences C and D define procedures for fine-tuning a fixed truss topology – joint repositioning or the adjustment of global members sizes, followed by the adjustment of the size of specific members. Sequence E describes a return from shape optimization to topology optimization – the repositioning of a joint (possibly to make room for new truss elements) followed by the addition of new structural members.

4.3 Results for Home Cooling System Design Study

The same methodology used to analyze the truss design data was applied to operation data from the cooling system design study. A plot of the log-likelihood for models of increasing order is provided in Figure 7(a). Many of the same trends from Figure 4(a) are echoed here.

There is an increase in log-likelihood between the zero-order and first-order Markov models, indicating that sequencing behavior is evident in the study data. There is a miniscule mean increase in testing accuracy between the first-order and second-order models, but this increase is nonsignificant ($F = 0.02,\ p > 0.5$). After second-order, the training log-likelihood decreases, again indicating that higher-order models tend to overfit the training data, losing generalizability. The marked divergence between training and testing curves displayed in Figure 7(a) (which was not as sharp in Figure 4) is indicative of the fact that higher-order models have a greater tendency to overfit this data set. This can be attributed to the fact that participants in the cooling system design study used far fewer operations than participants in the truss design study.

As with the analysis of the truss design study, the first-order Markov model is the preferred model for this design task. This may indicate that the sequencing of design operations can be treated as a first-order Markov process for the types of configuration tasks examined in this work, or perhaps more generally. At the very least, it is evidence that lower-order processes (but not zero-order) tend to be the most veridical. Higher-order models may learn specific sequential constructs that are informative, but they do not appear to offer a superior description of aggregate patterns of design activity.

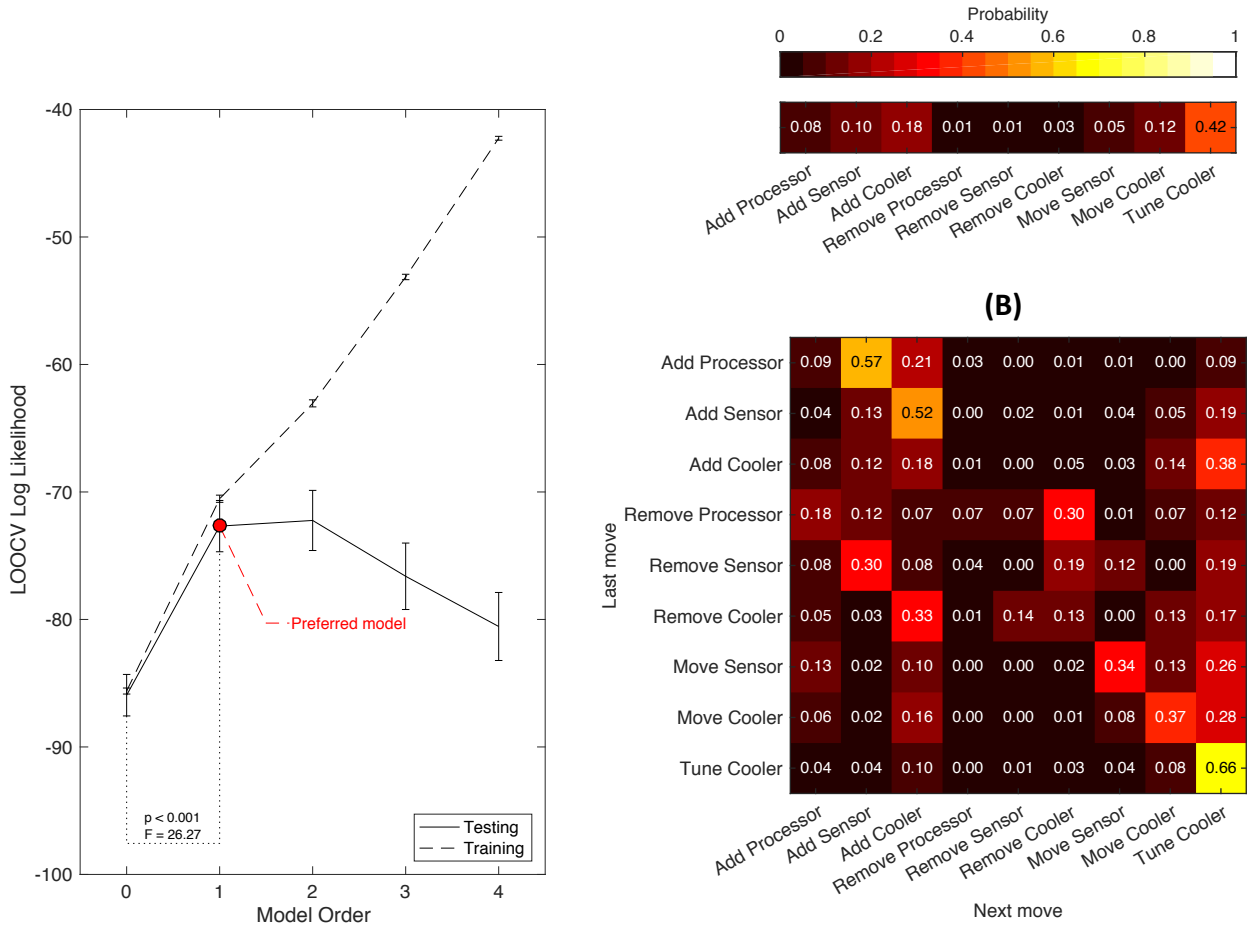

FIGURE 7. SUMMARY OF RESULTS FOR COOLING SYSTEM DESIGN STUDY, INCLUDING (A) LOG-LIKELIHOOD COMPUTED FOR MODELS OF INCREASING ACCURY, (B) STATE FREQUENCIES FOR ZERO- ORDER MARKOV MODEL, AND (C) TRANSITION MATRIX OF FIRST-ORDER MARKOV MODEL.

Examining the differences between Figures 7(b) and 7(c) can once again provide insight as to the benefit that is derived from pursuing operations sequentially for the cooling system design task. Whereas the first-order Markov model developed for the truss design data had a strongly diagonal structure, the transition matrix developed for the cooling system data has several strong off-diagonal elements and relatively weak diagonal elements. This indicates that there is little propensity to apply the same operator multiple times in a row. The only operations with more than a 30% chance of being applied multiple times in series are sensor movement, cooler movement, and cooler tuning. It should be noted that these are the shape operations. In

contrast, the topology operations (adding or removing products) have lower probabilities of being applied multiple times in series.

A graphical version of the first-order Markov transition matrix is provided in Figure 8 (thresholded in the same manner as Figure 5). This representation reinforces many of the trends observed in the raw transition matrix. Two trends are made particularly clear in this graph. The first is the highly probable linkage from adding a processor, to adding a sensor, to adding a cooler. This sequence enables the construction of the simplest independent subsystem possible, consisting of a sensor to read the temperature in a room, a cooler to act on the temperature in a room, and a processor to decide when to activate the cooler based on information from the sensor. The second trend is the strong connectedness of the cooler tuning operation to nearly every other operation. This indicates that cooler tuning played an integral role in the production of solutions, and was frequently utilized throughout the design process.

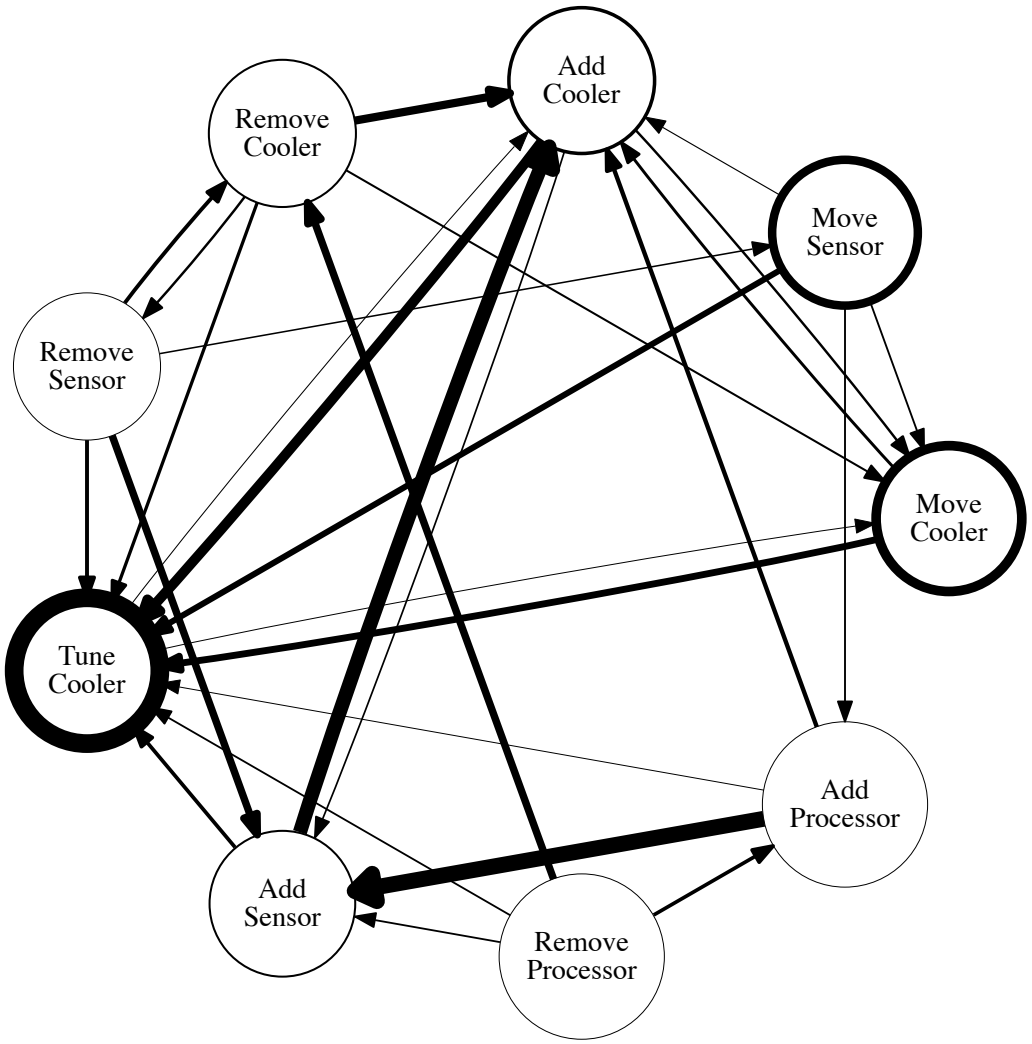

FIGURE 8. GRAPHICAL REPRESENTATION OF FIRST-ORDER MARKOV MODEL FOR THE COOLING SYSTEM DESIGN STUDY, SHOWING THE MOST LIKELY TRANSITIONS. TRANSITION PROBABILITY IS INDICATED BY THE LINE THICKNESS OF THE CORRESPONDING ARROW (IN THE CASE OF SELF-TRANSITIONS, THE THICKNESS OF THE BORDER OF THE CORRESPONDING OPERATION)

As shown in the results from the truss design study, longer sequences of operations can be extracted by traversing the graph-based representation of the transition matrix (see Figure **9**). Sequence A consists of a processor addition, a sensor addition, and a cooler addition. As noted above in Figure 8, this sequence encodes the construction of the simplest independent subsystem possible, consisting of a sensor, a cooler and a processor. Sequences B describes the removal followed by the addition of a cooler. These actions were necessary to transfer the control of a cooler to a different processor – this was not enabled with a single move during the study.

Interestingly, the probability of the opposite of these two actions (adding a cooler and then deleting a cooler) was nearly 0. Sequences C, D, and E are all sequences related to placing and modifying coolers. The prevalence of these sequences might be expected since the operations for adding and tuning coolers were applied the most often (see Figure 7(b)). Sequence C describes the common action sequence of adding a cooler and then immediately tuning its properties (a sequence employed by nearly every participant). An alternative cooler-related sequence is presented with Sequence D in which a cooler is added, moved to a new location, and then tuned. This sequence would have been enacted when the cooler did not function as expected where it was placed. Sequence E is related to sequence D in that it consists of the same operations but they are enacted in a different order. Sequence E begins with moving a cooler, an action that might leave part of the building under-cooled. A new cooler is then added (ostensibly in the under-cooled area) and tuned to optimize performance.

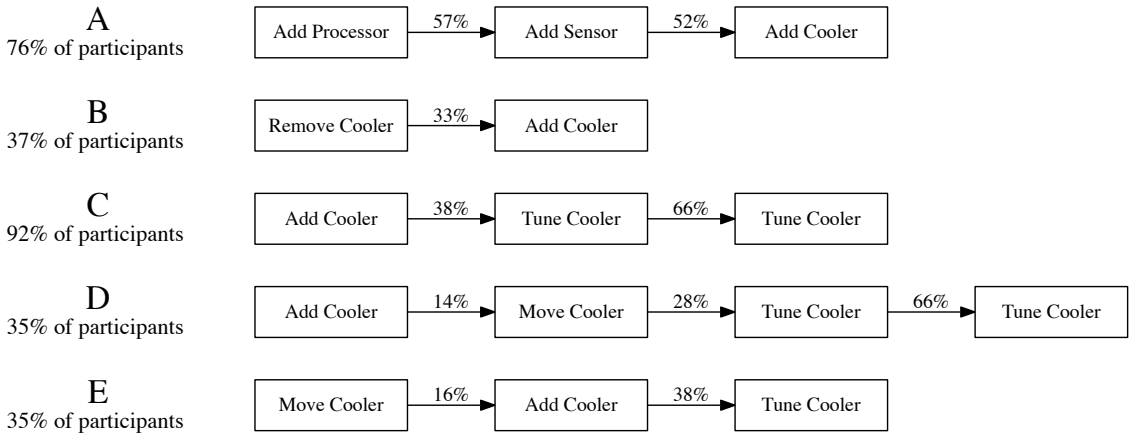

FIGURE 9. EXEMPLAR SEQUENCES EXTRACTED FROM FIRST-ORDER MARKOV MODEL.

4.4 Discussion

This section analyzed data from two design studies by fitting Markov models of increasing order to the operational data from the study. These models progressively encoded greater degrees of memory, meaning that the choice of the next operation in a sequence was based on knowledge of a greater number of prior operations. Two important findings resulted from this analysis.

1. It is likely that participants in both studies utilized operation sequences.

2. Designers' operational sequences can be modeled accurately using first-order Markov chains; higher-order Markov chains do not lead to significant increases in accuracy.

The first finding stems from a comparison of the zero- and first-order Markov models. Zero-order Markov models cannot encode sequence information, while first-order Markov models provide a minimal representation of sequencing, in which selection of the next operation is conditional upon only the last operation. For both studies, first-order Markov models fit the operation data better than zero-order models, thus demonstrating that operation sequencing is evident.

The second finding stems from a comparison of the first-order and higher-order Markov models. The first-order Markov model provided a fit that was either equivalent to or better than the higher-order models for both studies. Higher-order models encode sequences that are dependent on multiple prior operations, instead of just the most recent single operation. Therefore, the higher fit of the first-order model indicates that memory of multiple past operations is not necessary to accurately model the selection of future operations.

These first-order Markov models assume that a designers' choice of a subsequent operation is dependent only on what the last operation was, establishing a causal link between the two. By stringing together several of these causally linked operations, sequences of arbitrary length can be created. The process of creating these long sequences essentially amounts to traversing the graph described by the first-order transition matrix. As demonstrated in Figures 6 and 9, the longer sequences extracted by this method describe meaningful patterns of design that were employed often by study participants. These longer sequences might be represented more explicitly in higher-order Markov chains, but they are represented both succinctly and accurately using first-order Markov chains.

The number of independent parameters required to fully define the transition matrix for a Markov chain model is $(k-1)k^m$, where k is the number of possible operations and m is the order of the model. This means that the number of model parameters increases extremely rapidly

(exponentially) with the order of the model. As an illustration, consider the truss design problem which has 7 operations; a zero-order Markov chain requires the estimation of 6 independent parameters, a first-order model requires 42, and a second order model requires 294 parameters. The fourth-order model trained in this work required the estimation of more than 10,000 independent parameters. Larger numbers of parameters require larger quantities of training data in order to accurately estimate the values of the parameters. This may offer some intuition as to why the first-order models in this work were the most veridical in comparison to human data. Higher-order sequences simply require too much information, and are thus too burdensome to learn. This could heavily bias human problem-solvers towards lower-order sequences that can be learned with exponentially less information, enabling quick adaptation to new problems.

5 Investigation 2: Benefit of Operation Sequences

The first investigation established that there are recognizable operation sequences in the design data, and demonstrated that these sequences are best approximated using first-order Markov models. However, it has not been shown directly that sequence learning is beneficial to the solution of design problems. On one hand, the ability to learn sequences may help designers to learn and exploit problem-specific heuristics to quickly find fruitful regions of the design space. On the other hand, sequence learning may bias designers towards learning sequences that greedily improve solution quality. Applying these greedy sequences could critically limit breadth of search and lead designers towards local minima of inferior solution quality.

Because humans are capable of learning sequences implicitly [12,13] it is difficult to control and observe sequence-learning as an experimental variable in a study with human participants. It is possible that implicit learning processes could take over even if participants could be successfully prevented from explicitly learning any operation sequences. For that reason, this work utilizes the CISAT modeling framework [2] to test the effects of first-order sequence learning. Individuals and teams simulated in CISAT have clearly defined abilities, making it possible to perfectly control the way that these simulated individuals learn operational sequences. Therefore, a comparison between sequential and non-sequential learning patterns becomes feasible, and offers insight as to the benefit of sequence learning for real human designers.

5.1 Methodology

The CISAT modeling framework is an agent-based computational platform that is intended to simulate the process and performance of engineering design teams, and has been shown to do so accurately on the configuration-style design problems used in the current work [2]. At its core, the CISAT framework makes use of simulated annealing constructs. Eight additional cognitive characteristics are then layered on top of the core structure with the intention of supporting a more full description of how individuals proceed during design, and how they interact with one another as part of a team [2]. It should be noted that the ability to learn and employ sequences is a characteristic of individuals. However, the corpus of data used here (from both the truss and cooling system design tasks) was produced by individual designers operating within teams. Therefore, the CISAT simulations in this work are structured to simulate the performance of teams. The ability to learn sequences is implemented in CISAT at the agent level, reflecting the individual sequence-learning abilities of human designers.

CISAT agents learn how to apply operations through operational learning, one of the eight cognitive characteristics implemented in CISAT. The characteristic was originally implemented as follows. In every iteration, an agent selects which move operator to apply next by taking a random draw from a multinomial distribution defined by a vector of probabilities, $\boldsymbol{p}$. The chosen move operator, i, is then used to modify the current solution. If operator i improves the quality of the solution, then the probability that that operator will be chosen in the future is increased using the equation

$$p_i \leftarrow p_i \cdot (1 + k_{OL}), \tag{3}$$

where k_{OL} is a constant used to control how quickly learning occurs. If the move operator worsens the quality of the solution, the probability of selecting it in the future is decreased using

$$p_i \leftarrow p_i \cdot (1 - k_{OL}). \tag{4}$$

The probability vector is re-normalized following every update. Updating selection probability based on the effect of only the most recent application of a given move operator (instead of some

average over past applications) reflects availability bias [51], the tendency of humans to place greater weight on information that is readily available in memory.

A second version of the operational learning characteristic was implemented for this work using first-order Markov chain constructs. This model was chosen specifically because it was shown in the previous section to accurately encode human operation sequences. This concept is implemented so that agents select move operators to apply using the probabilities in a first-order Markov chain transition matrix, $\boldsymbol{T}$, and then iteratively tune the probabilities in $\boldsymbol{T}$ to bias selection towards the best sequences of move operators. This process is similar to the bipartite pattern/sequence generation process identified by Kotovsky et al. for human participants solving the Thurstone Letter Series Completion task [20]. Here, the tuning of probabilities in $\boldsymbol{T}$ constitutes the pattern generation step, and the selection of a subsequent operation based on $\boldsymbol{T}$ is analogous to the sequence generation step.

To select which move operator to apply, a random draw is taken from the multinomial distribution defined by row i of the matrix $\boldsymbol{T}$, where operator i is the last move operator applied. The chosen operator, operator j, is then applied to the current solution. The values in $\boldsymbol{T}$ are updated depending on whether the quality of the current solution improves:

$$T_{ij} \leftarrow T_{ij} \cdot (1 + k_{OL}), \tag{5}$$

or worsens:

$$T_{ij} \leftarrow T_{ij} \cdot (1 - k_{OL}). \tag{6}$$

Through the Markov chain transition matrix, the selection of the next move operator is made conditional upon the last move operator that was applied. This modification enables CISAT agents to learn and apply beneficial sequences of operations. Note that the first-order Markov chain model does not directly extract fixed-length sequences of operations from the experience of the agent. Rather, these sequences are implicitly coded within the transition matrix of the Markov chain model by updating probabilities according to move operator performance (i.e., whether applying a move operator improved or worsened solution quality). This makes the

enactment of previously beneficial sequences more likely, thus accounting for sequence learning within the CISAT modeling framework. However, the probabilistic nature of operator selection still allows for the exploration and discovery of new sequences. This is the only modification made to the framework, and all other constants and settings are as given in [2].

5.2 Results for Truss Design Study

In the interest of simplicity, performance was only simulated for the initial problem statement from the truss design study. A total of 100 teams were simulated for each of the two conditions: sequential, utilizing first-order Markov chain concepts; and non-sequential, utilizing zero-order Markov chains. The results of the simulations were then post-processed to track each team's best solution over time. The mean Normalized Strength-to-Weight Ratio (SWR) of each team's current best solution was then computed. This metric serves as an indication of the quality of a structural solution. A comparison of the two conditions is shown in Figure 10.

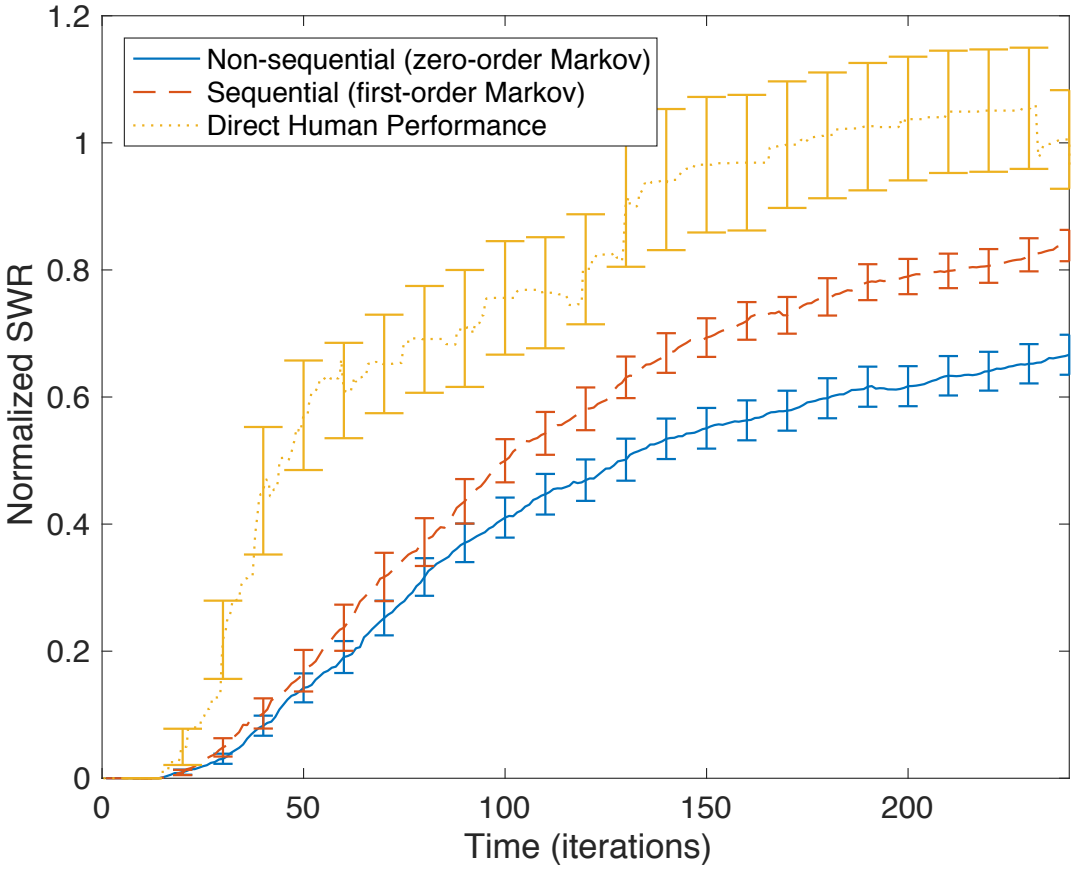

FIGURE 10. COMPARISON OF TRUSS DESIGN QUALITY FOR ZERO-ORDER MARKOV MODEL, FIRST-ORDER MARKOV MODEL, AND HUMAN PERFORMANCE ON THE TRUSS DESIGN PROBLEM (ERROR BARS SHOW ± 1 S.E.).

The difference in final design quality between the two simulated conditions (zero-order Markov and first-order Markov) is highly significant ($F = 11.2$, $p < 0.001$) and the condition using the first-order Markov chain learning approach achieved a higher final design quality. Since CISAT has been shown to accurately model engineering design teams, these simulations indicate that the ability to learn operation sequences while solving a design problem is beneficial to human designers. Although the introduction of sequence-learning abilities does not raise CISAT solution quality to the level of the real human teams, it closes the difference between the two by nearly half, indicating that the ability to learn sequences is vital to the success of real designers.

5.3 Results for Home Cooling System Design Study

CISAT was also used to simulate the performance of human design teams on the cooling system design task. Sequential and non-sequential learning was implemented as above with first-order and zero-order Markov chains, respectively. The results of the simulations were then post-processed to track each team's best solution over time. In this case, normalized cooling efficiency was computed for the series of best solutions as an indicator of quality. This metric is the cooling capacity of the system (the extent to which it decreased the peak temperature in the home), divided by the total cost of the system. This ratio was then normalized according to the target values for total cost and peak temperature. A comparison of the mean normalized cooling efficiency of the two conditions is shown in Figure 11.

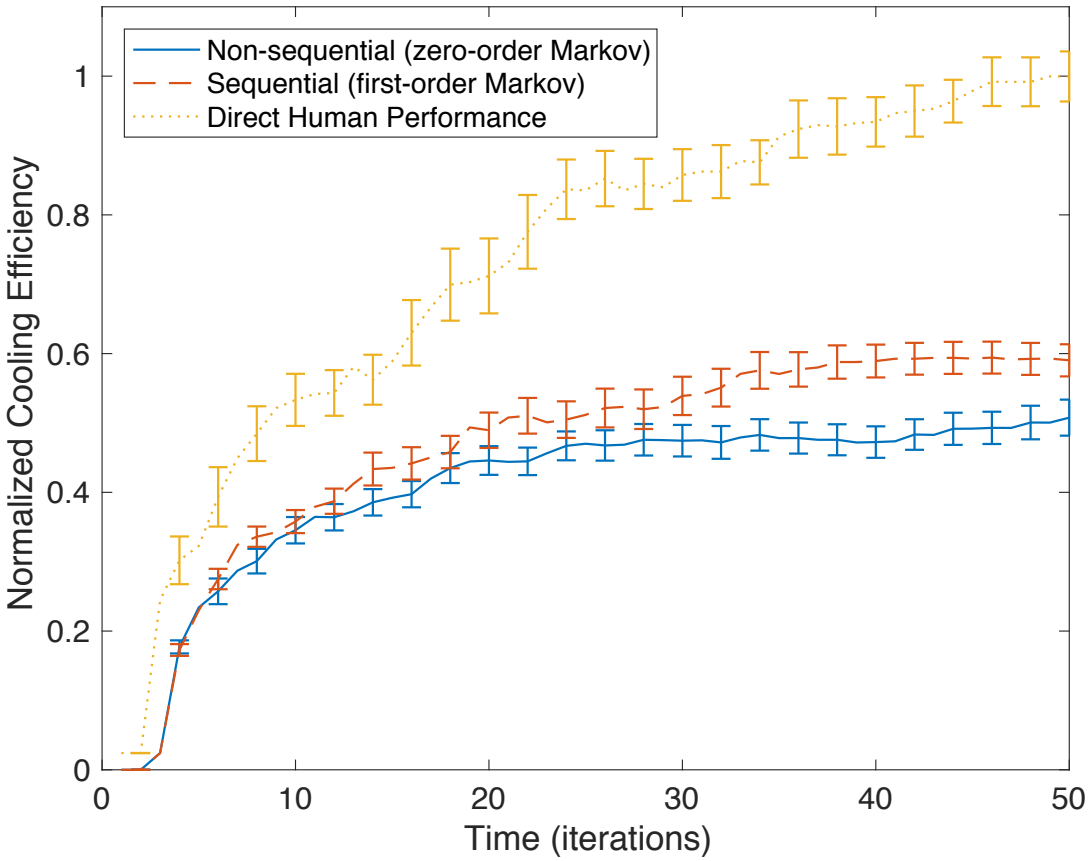

FIGURE 11. COMPARISON OF DESIGN QUALITY FOR ZERO-ORDER MARKOV MODEL, FIRST-ORDER MARKOV MODEL, AND HUMAN PERFORMANCE ON THE COOLING SYSTEM DESIGN PROBLEM (ERROR BARS SHOW ± 1 S.E.).

For this task, simulated teams that were capable of learning and employing sequences of operations achieved solutions with significantly higher quality ($F = 5.91,\ p < 0.05$). The increase in solution quality that results from the introduction of sequence-learning abilities again helps to close the gap between simulation and real human performance.

5.4 Discussion

The objective of this section was to assess whether or not the ability to learn sequences contributes positively to eventual solution quality. This was accomplished by modifying the operational learning characteristic of CISAT to enable agents to learn beneficial sequences of operations by reinforcing a first-order Markov transition matrix. This made it possible to perform a comparison between CISAT-simulated teams employing either non-sequential learning (a zero-order Markov model) or sequential learning similar to that observed in designers (a first-order Markov model). Simulations were conducted to reflect cognitive studies involving the design of trusses and the design of cooling systems, and in both cases sequential learning produced solutions with higher quality. This indicates that sequence learning is a beneficial aspect of human cognition during design.

The performance of sequence-learning and non-sequence-learning approaches is similar for the initial portion of the simulation on both design problems. A similar phenomenon was observed in other work that used machine learning to recognize and employ move operation pairs during computational design [38]. In that work, algorithms with and without the ability to learn pairs were compared, and showed nearly identical performance for the first 3% of the search. In that work, as well as the current paper, the identical early performance can be explained as an exploratory phase – the agent or algorithm is still learning about the design space, and has not yet learned effective move pairs or operation sequences. Once sufficient exploration has occurred, the agent or algorithm can begin to employ learned patterns to more effectively create solutions. This highlights the fact that, especially for sequence-learning, exploration and exploitation are inextricably linked. Human designers may stand to benefit from emphasizing the recognition of beneficial operation sequences during early exploration in order to aid more effective exploitation during the later stages of design.

6 Conclusions

This paper examined operation sequencing in engineering design using a variety of statistical and computational tools. Through analysis of human data from two cognitive studies, two research questions were specifically addressed:

1. *How much representational complexity is necessary to quantify the sequential patterns that design employ during solving?* Markov chain models with increasing order (representative of how much "memory" is assumed in the model) were fit to the data from two human studies. For both studies, the analysis indicated that the sequencing of design actions might be treated accurately as a first-order Markov process. It should be noted that longer finite-length sequences of operations may still be extracted by traversing the graphs described by the first-order Markov transition matrices.

2. *Is the employment of sequences of operations beneficial to designers?* Computational simulations were performed using the CISAT modeling framework to assess whether or not the ability to learn and employ operation sequences was beneficial to study participants. Several sets of simulations were conducted in which teams of agents solved the design problems from the two cognitive studies, either with the ability to learn sequences (encoded within a first-order Markov model) or without that ability (represented mathematically by a non-sequential statistical model). It was shown that the ability to learn operation sequences significantly increased the quality of solutions to both design problems.

The results of this work have the potential to inform novel approaches for design education and training. Here it was shown that designers utilized first-order sequences of operations, and that this allowed them to discover solutions with higher quality. Although is it not clear whether these sequences were learned implicitly or explicitly, there is evidence that explicit awareness of a sequence-learning task can improve performance [24]. Therefore, it is possible that teaching designers to be aware of the importance of learning sequences could improve their ability to learn said sequences. This self-awareness could be augmented in computer-aided design software by logging and analyzing design activity to provide real-time feedback about common sequential strategies, ensuring explicit awareness. Examining and quantifying the difference between explicit and implicit sequence-learning modalities with applications to design will be addressed in future work.

This work also holds implications for design automation and synthesis. Specifically, this work demonstrates that it may be possible to use Markov constructs to improve the effectiveness of design algorithms. The second investigation of this work implemented sequence-learning abilities in CISAT, a computational framework that is based in part on principles of stochastic optimization. This enabled computational agents to create and modify solutions using sequential chains of operations, resulting in improved performance. A similar implementation could be used to imbue other design synthesis algorithms with the ability to learn and apply sequences of operations. Such applications hinge on the fact that Markov chain models are generative [52], meaning that they encode the training data in such a way that they can be used to create new, synthetic data. In a design context, this amounts to the creation of new operational sequences that adhere probabilistically to observed patterns. Markov chain models could be learned and reinforced as an algorithm creates design solutions, or trained prior to use in an algorithm if sufficient prior data is available.

This work offers a foundation for describing sequence learning in engineering design by demonstrating that first-order Markov chains are a veridical model, and that learning simple first-order sequences can improve solution quality. These descriptive results establish a basis for future normative and explanatory research. Future explanatory work should investigate the underlying causation that gives rise to the sequential behaviors observed and described here, perhaps by using Markov Decision Process models [53]. Leveraging additional results from psychology (such as the importance of pauses during solving [54]) could also provide explanatory power. Further normative work could utilize numerical simulations to identify the dependence of learned sequences on the complexity and characteristics of the problem at hand by employing a predictive response surface methodology [47,55].

Acknowledgements

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE125252 and the United States Air Force Office of Scientific Research through grants FA9550-12-1-0374 and FA9550-16-1-0049. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors

and do not necessarily reflect the views of the sponsors. An early version of part of this work was included in the proceedings of the Design, Computing, and Cognition Conference [56].

References

[1] McComb, C., Cagan, J., and Kotovsky, K., 2015, "Rolling with the punches: An examination of team performance in a design task subject to drastic changes," Des. Stud., **36**(1), pp. 99–121.

[2] McComb, C., Cagan, J., and Kotovsky, K., 2015, "Lifting the Veil: Drawing insights about design teams from a cognitively-inspired computational model," Des. Stud., **40**, pp. 119–142.

[3] Neill, T. M., Gero, J. S., and Warren, J., 1998, "Understanding conceptual electronic design using protocol analysis," Res. Eng. Des., **10**(3), pp. 129–140.

[4] Kan, J. W., and Gero, J. S., 2011, "Comparing Designing Across Different Domains : an Exploratory Case Study," Design, (August).

[5] Yu, R., and Gero, J. S., 2015, "An empirical foundation for design patterns in parametric design," Proc. 20th Int. Conf. Assoc. Comput. Archit. Des. Res. Asia CAADRIA 2015, (April), pp. 1–9.

[6] Gero, J. S., and Peng, W., 2009, "Understanding behaviors of a constructive memory agent: A markov chain analysis," Knowledge-Based Syst., **22**(8), pp. 610–621.

[7] March, J. G., 1991, "Exploration and Exploitation in Organizational Learning," Organ. Sci., **2**(1), pp. 71–87.

[8] Gupta, A. K., Smith, K. G., and Shalley, C. E., 2006, "The Interplay Between Exploration and Exploitation," Acad. Manag. J., **49**(4), pp. 693–706.

[9] Goncher, A., Johri, A., Kothaneth, S., and Lohani, V., 2009, "Exploration and exploitation in engineering design: Examining the effects of prior knowledge on creativity and ideation," 2009 39th IEEE Frontiers in Education Conference, IEEE, pp. 1–7.

[10] Kotovsky, K., Hayes, J. ., and Simon, H. A., 1985, "Why are some problems hard? Evidence from Tower of Hanoi," Cogn. Psychol., **17**(2), pp. 248–294.

[11] Kotovsky, K., and Simon, H. A., 1990, "What makes some problems really hard: Explorations in the problem space of difficulty," Cogn. Psychol., **22**(2), pp. 143–183.

[12] Nissen, M. J., and Bullemer, P., 1987, "Attentional requirements of learning: Evidence from performance measures," Cogn. Psychol., **19**(1), pp. 1–32.

[13] Reed, J., and Johnson, P., 1994, "Assessing implicit learning with indirect tests: Determining what is learned about sequence structure.," J. Exp. Psychol. Learn. Mem. Cogn., **20**(3), pp. 585–594.

[14] Clegg, B. A., Digirolamo, G. J., and Keele, S. W., 1998, "Sequence learning," Trends Cogn. Sci., **2**(8), pp. 275–281.

[15] Chase, W. G., and Simon, H. A., 1973, "Perception in Chess," Cogn. Psychol., **4**(1), pp. 55–81.

[16] Egan, D. E., and Schwartz, B. J., 1979, "Chunking in recall of symbolic drawings," Mem. Cognit., **7**(2), pp. 149–58.

[17] Reitman, J. S., 1976, "Skilled perception in Go: Deducing memory structures from inter-response times," Cogn. Psychol., **8**(3), pp. 336–356.

[18] Moss, J., Cagan, J., and Kotovsky, K., 2004, "Learning from design experience in an agent-based design system," Res. Eng. Des., **15**, pp. 77–92.

[19] Pretz, J. E., Naples, A. J., and Sternberg, R. J., 2003, "Recognizing, Defining, and Representing Problems," The Psychology of Problem Solving, J.E. Davidson, and R.J. Sternberg, eds., Cambridge University Press.

[20] Simon, H. A., and Kotovsky, K., 1963, "Human acquisition of concepts for sequential patterns," Psychol. Rev., **70**(6), pp. 534–546.

[21] Kotovsky, K., and Simon, H. A., 1973, "Empirical tests of a theory of human acquisition of concepts for sequential patterns," Cogn. Psychol., **4**(3), pp. 399–424.

[22] Novick, L. R., and Tversky, B., 1987, "Cognitive constraints on ordering operations: the case of geometric analogies.," J. Exp. Psychol. Gen., **116**(1), pp. 50–67.

[23] Perruchet, P., and Amorim, M.-A., 1992, "Conscious Knowledge and Changes in Perfomance in Sequence Learning: Evidence Against Dissociation," J. Exp. Psychol. Learn. Mem. Cogn., **18**(4), pp. 785–800.

[24] Willingham, D. B., Nissen, M. J. J., and Bullemer, P., 1989, "On the development of procedural knowledge," J. Exp. Psychol. Learn. Mem. Cogn., **15**(6), pp. 1047–1060.

[25] Curran, T., and Keele, S. W., 1993, "Attentional and Nonattentional Forms of Sequence Learning," J. Exp. Psychol. Learn. Mem. Cogn., **19**, pp. 189–202.

[26] Stempfle, J., and Badke-schaub, P., 2002, "Thinking in design teams - an analysis of team communication," Des. Stud., **23**(5), pp. 473–496.

[27] Atman, C. J., Adams, R. S., Cardella, M. E., Turns, J., Mosborg, S., and Saleem, J., 2007, "Engineering Design Processes: A Comparison of Students and Expert Practitioners," J. Eng. Educ., **96**(4), pp. 359–379.

[28] Goldschmidt, G., and Rodgers, P. A., 2013, "The design thinking approaches of three different groups of designers based on self-reports," Des. Stud., **34**(4), pp. 454–471.

[29] Radcliffe, D. F., and Lee, T. Y., 1989, "Design methods used by undergraduate engineering students," Des. Stud., **10**(4), pp. 199–207.

[30] Todd, D., 1997, "Multiple Criteria Genetic Algorithms in Engineering Design and Operation," University of Newcastle.

[31] Rogers, J., 1996, "DeMAID/GA - An enhanced design manager's aid for intelligent decomposition," 6th Symposium on Multidisciplinary Analysis and Optimization, American Institute of Aeronautics and Astronautics.

[32] Sen, C., Ameri, F., and Summers, J. D., 2010, "An Entropic Method for Sequencing Discrete Design Decisions," J. Mech. Des., **132**(10), p. 101004.

[33] Waldron, M. B., and Waldron, K. J., 1988, "A time sequence study of a complex mechanical system design," Des. Stud., **9**(2), pp. 95–106.

[34] Meier, C., Yassine, A. A., and Browning, T. R., 2007, "Design Process Sequencing With Competent Genetic Algorithms," J. Mech. Des., **129**(6), pp. 566–585.

[35] Gero, J. S., 1990, "Design Prototypes : A Knowledge Representation Schema for Design," AI Mag., **11**(4), pp. 26–36.

[36] Bhatta, S. R., and Goel, A. K., 1994, "Discovery of physical principles from design experiences," Int. J. AI EDAM (AI Eng. Des. Anal. Manuf., (July 1992), pp. 1–22.

[37] Gero, J. S., Kan, J. W., and Pourmohamadi, M., 2011, "Analysing Design Protocols : Development of Methods and Tools," Res. into Des., pp. 3–10.

[38] Vale, C. A. W., and Shea, K., 2003, "A Machine Learning-Based Approach to Accelerating Computational Design Synthesis," Proc. 8th Int. Conf. Eng. Des., pp. 445–446.

[39] Stroock, D. W., 2005, An Introduction to Markov Processes, Springer Science and Business Media.

[40] Markov, A. A., 1907, "Extension of the Limit Theorems of Probability Theory to a Sum of Variables Connected in a Chain," The Notes of the Imperial Academy of Sciences of St. Petersburg, VIII Series, Physio-Mathematical College XXII.

[41] Scherr, A. L., 1962, "An Analysis of Time-Shared Computer Systems," Massachusetts Institute of Technology.

[42] Page, L., Brin, S., Motwani, R., and Winograd, T., 1999, The PageRank Citation Ranking: Bringing Order to the Web., Stanford InfoLab.

[43] Tamir, A., 1998, Applications of Markov Chains in Chemical Engineering, Elsevier.

[44] Gero, J. S., and Kan, J. W., 2009, "Learning to collaborate during team designing: some preliminary results from measurement-based tools," ICORD 09: Proceedings of the 3rd International Conference on Research into Design Engineering, Bangalore, India, pp. 560–567.

[45] Gero, J. S., and Kan, J. W., 2011, "Learning to collaborate during team designing: quantitative measurement," ICORD 11: Proceedings of the 3rd International Conference on Research into Design Engineering, pp. 978–981.

[46] Raftery, A. E., 1985, "A Model for High-Order Markov Chains," J. R. Stat. Soc. Ser. B, **47**(3), pp. 528–539.

[47] McComb, C., Cagan, J., and Kotovsky, K., 2017, "Optimizing Design Teams Based on Problem Properties: Computational Team Simulations and an Applied Empirical Test," J. Mech. Des., **139**(4), p. 41101.

[48] Arlot, S., and Celisse, A., 2010, "A survey of cross-validation procedures for model

selection," **4**, pp. 40–79.

[49] Stone, M., 1974, "Cross-validatory choice and assessment of statistical predictions," J. R. Stat. Soc. Ser. B, **36**(2), pp. 111–147.

[50] Hastie, T., Tibshirani, R., and Friedman, J., 2009, The Elements of Statistical Learning, Springer New York, New York, NY.

[51] Tversky, A., and Kahneman, D., 1973, "Availability: A heuristic for judging frequency and probability," Cogn. Psychol., **5**(2), pp. 207–232.

[52] Bishop, C. M., and Lasserre, J., 2007, "Generative or discriminative? getting the best of both worlds," Bayesian Statistics, J.M. Bernardo, M.J. Bayarri, J.O. Berger, A.P. Dawid, D. Heckerman, A.F.M. Smith, and M. West, eds., pp. 3–24.

[53] Bellman, R., 1957, "A Markovian Decision Process," J. Math. Mech., **6**(5), pp. 679–684.

[54] Chi, M. T. H., Glaser, R., and Rees, E., 1982, "Expertise in problem solving," Advances in the Psychology of Human Intelligence, Erlbaum, Hillsdale, NJ, pp. 7–75.

[55] Dinar, M., Park, Y.-S., Shah, J. J., and Langley, P., 2015, "Patterns of Creative Design: Predicting Ideation From Problem Formulation," Volume 7: 27th International Conference on Design Theory and Methodology, ASME, p. V007T06A006.

[56] McComb, C., Cagan, J., and Kotovsky, K., 2016, "Utilizing Markov Chains to Understand Operation Sequencing in Design Tasks," Design Computing and Cognition '16, J.S. Gero, ed., Springer International Publishing, Cham, Switzerland, pp. 401–418.