Dynamic Structuring in Cellular Self-Organizing Systems

Newsha Khani and Yan Jin
University of Southern California, USA

Conventional mechanical systems composed of various modules and parts are often inherently inadequate for dealing with unforeseeable changing situations. Taking advantage of the flexibility of multi-agent systems, a cellular self-organizing (CSO) systems approach has been proposed, in which mechanical cells or agents self-organize themselves as the environment and tasks change based on a set of rules. To enable CSO systems to deal with more realistic tasks, a two-field mechanism is introduced to describe task and agents complexities and to investigate how social rules among agents can influence CSO system performance with increasing task complexity. The simulation results of case studies based on the proposed mechanism provide insights into task-driven dynamic structures and their effect on the behavior, and consequently the function, of CSO systems.

Introduction

Adaptability is needed for systems to operate in harsh and unpredictable environments where it is impossible for the designer to conceptualize every possible incident and predict details of changing functional requirements. Space and deep sea explorations and rescue missions in hazardous environments are some examples of such variable environments. In most, if not all, engineered systems, the physical components are designed for a limited purpose and restricted operation range, beyond which the behaviors are not predictable.

The existing approach to dealing with changing task environments relies on designers' imagination of a variety of possible situations of the task domain that helps them devise needed responses to the imaginable possi-

Design Computing and Cognition DCC'14. J.S. Gero (ed),
pp. xx-yy.

bilities. Following the law of requisite variety [1]—i.e., only variety (of the system) can conquer variety (of the task)—this approach increases the system variety by adding more components and therefore enlarging system state space. While the approach has been effective for many complex systems developed to date, as the system components become too many, highly sophisticated and their interactions more intertwined, unintended interactions will ensue, making it difficult for designers to ensure the valid operation range for the system to survive its expected lifecycle.

As an alternative approach to adaptive and complex engineered systems, a cellular self-organizing (CSO) systems approach has been proposed [2][3][4]. Like many other multi-agent systems, A CSO system is composed of multiple homogeneous or heterogeneous mechanical cells (mCells, i.e., agents) that can be a small functional component or a robot. Each mCell is equipped with needed sensors and actuators and encoded with system design-DNA (dDNA) containing the information that specifies cellular actions and decisions for taking these actions. mCells interact with their task environment and with each other, leading to self-organizing emergent behavior and functions at the system level. To facilitate mCells' interactions with the task environment, a task field-based regulation (FBR) mechanism has been developed [4]. To explore mCell interactions, a COARM (collision, avoidance, alignment, randomness, and momentum) parametric model has been examined [3].

The self-organizing behavior of the current CSO systems is regulated by each mCell transforming the task environment into a task-field in which it finds its "most comfortable place" and moves into it. The task is completed by the collective effort of the mCells making themselves "more comfortable." Each mCell makes their movement decisions completely based on its own sensed information of environment, its own transformation algorithm, and its own decisions for action. mCells collectively perform the task by first "discovering" what the task is (where is the "comfortable place") and then "carrying out" the task (move into the "comfortable place"). This distributed and self-interested approach allows for flexibility to cope with changing tasks, robustness to deal with changing environment, and resilience to still function with system dismemberment.

The current field-based regulation (FBR) approach to self-organization has two problems. First, when the task becomes more complex, both the description and transformation of task-field become highly complicated, potentially becoming a design hurdle. Second, the current approach does not directly address the interaction between mCells with respect to the task, leaving the power of mCells' self-organized structures unutilized. As will be described in the following sections, the problems have become evident when we make the box-moving task to include "rotate" the box in ad-

dition to simply "push/move" the box. This increased complexity of the task has made the simple FBR based CSO system incapable of completing the task in most cases, even when the field description is fully supplied. There is a need to incorporate task-driven dynamic structuring among mCells into the self-organizing framework.

Social structures play an important role in solving collective tasks. Many complex systems are hierarchical in structure including social systems, biological systems and physical systems [5]. Structures can be found everywhere in society, such as governments, companies, and universities. Many natural systems, over eons of time, have participated in the evolution process to organize themselves into a more complex and favorable arrangement [6].

In this research, we explore a dynamic social structuring approach to enhance the self-organizing functionality for CSO systems. We attain social structuring among mCells by introducing both general and context-based social rules and devise a social-rule based regulation (SRBR) for mCells to choose their actions. To facilitate SRBR, we introduce the concept of "social field" in addition to the current "task field." In SRBR, mCells' behavior is adjusted through perceived social field to be in harmony with system-wide welfare. Social rules can be designed based on the task definition and resolution of possible occurring conflicts.

In the rest of this paper, we first review the related work in Section 2, and then, in Section 3, introduce our dynamic social structuring concepts and present social rule based behavior regulation (SRBR) approach. In Section 4 we demonstrate the effectiveness of our approach through simulation-based case studies. Section 5 draws conclusions and points to future research directions. In the following, we will use the word "agent" and "mCell" interchangeably and will use the latter only when necessary for emphasizing CSO features.

Related Work

In the field of engineering design, design for adaptability and design of reconfigurable systems have been investigated in the past decade. In their work focusing on vehicle design, Ferguson and Lewis [7] introduced a method of designing effective reconfigurable systems that focuses on determining how the design variables of a system change, as well as investigating the stability of a reconfigurable system through the application of a state-feedback controller. Martin and Ishii [8] proposed a design for variety (DFV) approach that allows quick reconfiguration of products but main-

ly aims to reduce time to market by addressing generational product variation. Indices have been developed for generational variance to help designers reduce the development time of future evolutionary products. In addition to developing design methods for reconfigurable systems, various reconfigurable robotics have been developed mostly by computer scientists. Unsal et al [9] focused on creating very simplistic i-Cube systems (with cubes being able to attached to each other) in order to investigate whether they can fully realize the full potential of this class of systems. PolyBot has gone through several updates over the years [10] but acquired notoriety by being the first robot that "demonstrated sequentially two topologically distinct locomotion modes by self-configuration. SuperBot [11] is composed of a series of homogeneous modules each of which has three joints and three points of connection. Control of SuperBot is naturally inspired and achieved through a "hormone" control algorithm.

Despite the implicit and informal nature of some multi-agent relations, all multi-agent systems possess some form of organization. For a distributed system with the purpose of solving a problem or reaching objective functionality, an organized way of sharing information among agents can be very helpful. Organizational oriented design has shown to be effective and is typically used to achieve better communication strategies [12]. It has been proved that the behavior of the system depends on shape, size and characteristics of the organizational structure [13][14]. Researchers have suggested that there is no single type of organization that is a best match for all circumstances [13].

As an alternative approach to adaptive and complex engineered systems, the previous work on cellular self-organizing systems (CSO) has provided useful insights into understanding necessary characteristics of adaptive systems and introducing nature inspired concepts. The current FBR approach is fully distributed since every mCell works on their own without considering other mCells. From a multi-agent system's perspective, the full distribution represents a level of disorderliness that has two important implications. First, the disorderliness means limited functional capabilities because the system lacks ways to create corresponding sophistication when tasks become more complex. Second, the disorderliness, on the other hand, provides an opportunity for us to infuse order into the system and therefore increase the level of overall system capability. The question is *how can we devise such order so that we can "control" the level of orderliness for best balance of system adaptability and functionality*?

A Social Rule Based Regulation Approach to Dynamic Social Structuring

Basic Idea

As mentioned above, a system needs to possess a certain level of complexity in order to deal with tasks with a corresponding level of complexity [1]. Furthermore, it has been demonstrated that a system with higher physical complexity is more adaptable because the higher-level diversity permits satisfaction of changes of constraints around the system [15]. Although algorithmic information content based complexity measure equals randomness with complexity, from a system design perspective, it is more appropriate to count the complexity of a system based on its physical, structural, and effective features. In this case, pure randomness is discounted and the attention is placed on agent interactions and evolving structures.

Following Huberman and Hogg [15], we consider the complexity spectrum of engineered systems over order and disorder bell shaped, as illustrated in Fig. 1. A single solid object, such as a hammer, has complete order, as indicated in point (a) in Fig. 1; it has close to zero complexity and can deal with very simple tasks, such as punching a nail. By increasing number of dedicated components and introducing interactions between them, the order decreases in the sense that the system can be in various ranges of possible states. Such systems can be a gearbox (simpler) or an internal combustion engine (more complex). Although this "complexity by design" approach (from (a) to (c) in Fig. 1) has been the mainstream approach to complex engineered systems and has been highly effective, the unintended and unknown interactions among the sophisticated components may potentially become a "showstopper" when the systems demand super complexity for super demanding tasks. Space mission accidents and those of nuclear power plants are examples.

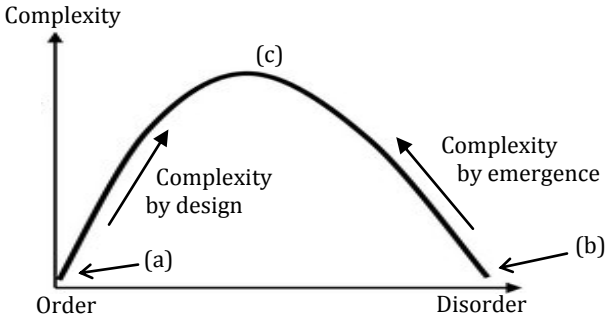


Fig. 1 Hypothetical system complexity over order-disorder spectrum (adapted from [15])

An alternative approach to complex engineered systems is to start from completely disorganized simple agents (or mCells in our CSO term), as indicated by point (b) in Fig. 1. While the completely disordered agents cannot perform any task, not even punching nails, introducing order among the agents can potentially lead to a functional system (moving from (b) to (c) in Fig. 1). Physical materials, biological systems, and ant colonies are examples. The distinctive feature of this approach to complex engineered system is "complexity by emergence." Since "by emergence" does not require explicit knowledge of specific interactions among agents, the "show-stopper" mentioned above can be avoided. Furthermore, this approach may fundamentally expand the design of engineered systems by bringing biological developmental concepts into mechanical system development.

Our research on self-organizing systems takes the "by emergence" approach. Besides introducing the concepts of design-DNA (dDNA) and mechanical cell (mCell, i.e., agent), a task field based behavior regulation (FBR) mechanism has been developed to allow agents to self-organize (i.e., introducing order) through each agent seeking attractors of its perceived task field. Based on Fig 1, previous CSO system designs fall into a cluster of points close to (b). Although this limited orderliness was effective for completing "push box" tasks, it was not enough for "push and rotate box." To further increase the level of orderliness, in this research we introduce the concept of "social structure" to capture explicit interactions among agents and apply "social rules" to facilitate dynamical social structuring among agents.

In the following subsections, we first introduce the measure of task complexity and then describe the models of agents, social structures, and social rules, followed by the social rule based regulation mechanism. The subsequent case study sections will demonstrate how higher level task complexity demands dynamic structuring and how social rule-based regulation can be applied to increase the orderliness, and consequently the capability, of the overall system.

Task Complexity

In the CSO framework, tasks together with environmental situations are presented as "task fields" in which agents seek and move to attractors [4]. Task field is defined as below.

Definition 1 (Task Field & Field Formation): $tField = FLD_t(FR, ENV)$
where, FLD_t: field formation operator, FR is set of functional requirements of task, and ENV is set of environment constraints.

At a given time, an agent's behavior can be self-regulated based on its "*field position*" at that moment which is determined by the task requirements and the environmental situation. We assume that each agent is equipped with needed sensors and a field formation operator.

To demonstrate that more complex tasks require more complex systems, a measure of task complexity is needed. Various measures of task complexity have been proposed [16][17]. We define task complexity to have four components: composite complexity, object complexity, coordination complexity and dynamic complexity.

Tasks are composed of various functions. Typically, a function can be represented as a pair of <verb> <object> (e.g., <push><box>). As the number of distinguishable verbs (i.e., actions) of a task increases, agents need to be more knowledgeable in order to perform the task. Therefore, the number of distinct verbs can be used as a measure of an aspect of task complexity, called *composite complexity*. We have,

Definition 2 (Composite Complexity): $CoC = \sum_{i=1}^{|V|} 1 + \sum_{i=1}^{|O|} 1$

where, $V=\{v_1, \ldots, v_n\}$ *is the set of all distinguished actions and* $O=\{o_1, \ldots, o_m\}$ *is the number of all distinguished objects;* $|V| = n$ *and* $|O| = m$.

In addition to the number of objects, the properties of the objects involved in a task, such as shape, dimension, and mass, also contribute to the task complexity. Therefore the number of parameters used to describe the distinctive objects can be used to define the *object complexity* of the task. The more the parameters are, the higher the complexity level is. We define object complexity of a task as,

Definition 3 (Object Complexity): $ObC = \sum_i^N P_i$

where, N is number of unique objects involved in the task, P_i *is number of parameters for describing object i.*

For a given task, in addition to the number of actions, there can be various relationships between these actions that must be maintained for the completion of the task. Examples include timing between actions (e.g., parallel, sequential, or specific delay) and number of relative occurrences. The existence of these relationships requires coordination of actions within an agent and between multiple agents. This change in phase of agent action can add a fair amount of *coordination complexity* to the system.

Definition 4 (Coordination Complexity): $CoC = \sum_{i \in V} \sum_{j \in V} r_{ij}$

where, r is action relations between action (verb) i and j, which can be sequential or reciprocal .

Another kind of task complexity deals with the changing environment. When environment changes, task field will vary. Depending on the degree of variation, an agent's behavior for action and coordination should be adjusted. We can capture such *dynamic complexity* by the sum of differences across a certain time period for the abovementioned three complexity components, as described in [16]. We have,

Definition 5 (Dynamic Complexity):

$$DyC = \left|CpC_{(t+1)} - CpC_{(t)}\right| + \left|ObC_{(t+1)} - ObC_{(t)}\right| + \left|CoC_{(t+1)} - CoC_{(t)}\right|$$

The overall task complexity is the weighted sum of the abovementioned complexities:

Definition 6 (Task Complexity):

$$TC = W_{cp}CpC + W_{ob}ObC + W_{co}CoC + W_{dy}DyC$$

where, W_{cp},W_{ob}, W_{co}, W_{dy} are the weights assigned to each complexity measure.

Examples of how these complexity measures are applied and computed are given in the case study section.

Agent and Social Structures

In the CSO framework, we treat mechanical components as mechanical cells (mCell, i.e., agents). Following our previous work [4], we have following definitions:

Definition 7 (Mechanical Cell): mCell = {Cu, S, A, B};
where Cu: control unit; S = {s_1, s_2, ...}: sensors/sensory information; A = {a_1, a_2, ...}: actuators/actions; B: designed behavior, or design information (see definition 4 below).

Mechanical Cell is the smallest structural and functional unit of a CSO system. Although for a CSO system design, the appearance or the structure of its *mCells* may be different, a *mCell* should be able to sense the environment and process material, energy and/or information as their actions.

Definition 8 (State): State = {S_C, A_C}
where $S_C \subset S$ and $A_C \subset A$ are currently sensory information and actions, respectively.

State is used to represent the situation. It is the combination of the current sensor information *Sc* and current actions *Ac.*

Definition 9 (Behavior): $b = \{S_E, A_E\} \rightarrow A_N$

where $S_E \subset S$ and $A_E \subset A$ are existing sensor information and actions, respectively; and $A_N \subset A$ are next step actions.

A behavior *b* is the designed action for given situations or states. The *Cu* of the *mCell* should be able to judge the situation and make decisions on next actions. The design information of a CSO system is the fully developed behaviors for each *mCell*.

One important feature of a CSO system is that each agent is self-interested; they always seek attractions (i.e., attractors) that make them "happier." It is this self-organizing behavior that makes the overall system robust and adaptive to change. However, such self-organizing behavior must be effectively guided so that structures, and therefore complexity, can emerge and the overall system can be functional. In this research, the notion of *satisfaction* is used to capture the happiness of an agent in choosing their actions. For a single agent without considering the existence of other agents, its satisfaction can be defined as below:

Definition 10 (Agent Satisfaction): $Sat_{Beh_i} = Eff(Beh_i, tField)$

where, Beh_i ={beh_1,…,beh_n} set of behaviors available to agent i, Eff returns effectiveness profile of Beh_i in the current tField.

An individual agent's satisfaction is a function that maps the all available behaviors to the effectiveness with respect to its task field. The values of an agent's satisfaction for each possible behavior in the current task field constitute a profile of probabilities of executing for all possible behaviors. This is identical with the FBR described in [4].

Along the similar line of thinking about task complexities mentioned above and by focusing on the physical and effective features [15][18], we consider the complexity of an agent in terms of the agent's number of actions, number of behaviors and communication capacity (e.g., range and number of channels). We have,

Definition 11 (Individual Agent Complexity): $C_{agent_i} = N_a + N_b + C_{Com}$

where, N_a is the number of actions, N_b is the number of behaviors , C_{com} is the communication capacity.

Definition 12 (System's Agents Complexity): $C_{agents} = \sum_{i=1}^{N} C_{agent_i}$

where, N is the number of agents.

To increase the emergent complexity and level of sophistication of a multi-agent system requires devising order into the system, as indicated in Fig. 1. In this research, we devise order by introducing social structures among agents. More specifically, we apply the graph theory principles to capture the interactions among agents.

Assume *G* is a set of all possible graphs that can be formed by N agents $Ag = \{a_1, a_2, \ldots, a_N\}$. Then we define

Definition 13 (Social Structure): G(t) = (N, E(t)),
where, N is the number of agents, N is the number of agents , E(t) is the links of interactions/relations between agents at time t.

As shown above, social structure *G(t)* is a function of time and is directly dependent on the evolution of agents' interactions. For simplicity, we assume agents are constant nodes in the graph while edges between the nodes changes over time resulting in a dynamic structure.

In CSO systems, the social structure represented as connectivity graph is realized by defining social rules that specify how agents interact with each other. These social rules can be general (e.g., "move to similar direction with neighbors) or task specific (e.g., "move closer with neighbors in on the edge of a box"). We define social complexity measure of agents based on their connectivity graph that originates from social rules. This type of graph complexity is notably similar to the complexity measures defined in molecular chemists [19]. The vertex degree magnitude-based information content, I_{vd} is based on Shannon entropy and defines information as the reduced entropy of the system relative to the maximum entropy that can exist in a system with the same number of elements. The analysis has shown that the I_{vd} index satisfies the criteria for a measure of network complexity. It increases with the connectivity and other complexity factors, such as the number of branches, cycles, cliques, etc.

Definition 14 (Social Complexity): $\mathrm{SC} = \sum_{i=1}^{N} d_i \log(d_i) / N$
where, d_i is the degree of each node i (how many other agents are communicating with agent i).

Social Rule Based Behavior Regulation

The main objective of this research is to explore ways to facilitate emergence of order and therefore complexity so that a CSO system can deal with more complex tasks. We want to devise dynamic structuring methods that can help guide agents to self-organize. We take a social rule based behavior regulation approach and explore various local and bottom up social relations to achieve dynamic social structuring.

Generally speaking, the deficiency of disorderliness or disorganization can be divided into two categories. One is "conflict deficiency" and the other "opportunity-loss deficiency." For simple tasks (e.g., push a box to a destination in an open space) where individual agent's "goal" is mostly consistent with the system goal, the agents' effort can additively contribute to the system overall function. When tasks become more complex, con-

flicts between agents' actions (e.g., push box in opposite directions due to space constraints) may occur and cooperation opportunities may be lost.

In order to minimize the conflict between agents and exploit cooperation opportunities, social rules and social relations can play an important role. A social rule is a description of behavioral relationship between two encountering agents that can be used by the agents to modify their otherwise individually, rather than socially, determined actions. Two agents acting on a give social rule are said to be engaged in a social relationship. Based on *definition 13* mentioned above, when agents are engaged in social relations by following social rules, social structures emerge, leading to more order and higher complexity of the system.

To avoid conflicts and promote cooperation, social rules can be defined to specify which actions should be avoided and which actions are recommended for given conditions. The conditions are often task domain dependent, although they can also be general. We have,

Definition 15 (Social Rule): sRule = <C, ForA, RecA>
where C is a condition specifying a set of states; ForA: forbidden actions for states specified by; RecA: suggested action.

Social rules defined above introduce relations among encountering agents. It is conceivable that when an agent encounter neighbors and neighbors encounter their neighbors the cascading effect may lead to a large scale network structure with varying densities. The distribution of such densities can be defined as a *social field* in which every agent has its own position and the awareness of the social field allows agent to reach (i.e., be aware of) beyond the encountering neighbor agents. We have,

Definition 16 (Social Field): sField = FLDs (sRule)
where FLDs is the field formation operator; sRule is a social rule.

Social field adds another layer to the design of CSO systems as a helpful mechanism to secure unity in the system. We will explore its effect in future research. In this research, the focus is put on allowing agents to adjust their otherwise individual satisfaction behavior (see *definition 10*), based on applying social rules to the encountering neighbor agents. This social rule based behavior regulation (SRBR) can be defined as follows.

Definition 17 (Social Rule Based Behavior Regulation):

$$SocSat_{Beh_i} = SRBR(Sat_{beh_i};\ SR_i, NA_i)$$

Where, SRBR is social field based regulation operator for behavior correction; Sat_{beh_i} *is tField based behavior satisfaction (see definition 10);* SR_i *is set of social rules;* NA_i *is set of encoun-*

tering neighbor agents; $SocSat_{Beh_i}$ *is socially regulated behavior satisfaction.*

The above is a general definition. To apply SRBR, an agent needs to 1) generate its independent satisfaction profile through FBR (see *definition 10*), 2) identify and communicate with its neighbors, 3) possess social rules, 4) know which rule to apply for a given situation, and 5) know how to generate new social satisfaction behavior. Each of the 5 steps can be task domain dependent. In the following section, we discuss how these steps can be implemented and the above mentioned concepts be applied.

Case Study

The objective of case study is to explore and demonstrate how social rule based behavior regulation can increase the order, and therefore the complexity, of the overall system and how this increased order is essential for dealing with more complex tasks. To pursue this objective, we developed a multi-agent simulation system based on the NetLogo platform [20], a popular tool used by researchers of various disciplines.

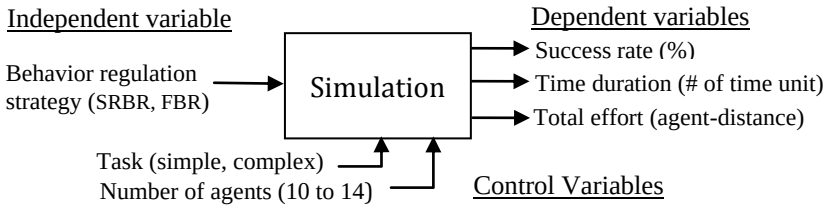


Fig. 2 Experiment design

Fig. 2 illustrates the design of simulation based experiment. As independent variable, two strategies were explored, with social structuring (SRBR), and without social structuring (FBR). Control variables are used to test different task and agent situations. Two tasks were tested, pushing a box without an obstacle (simple) and pushing a box with an obstacle (complex). For all settings, we measure success rate, time duration (number of steps) and total effort (total distance the agents traveled) as dependent variables.

Tasks

The box-moving task used for the cases study is illustrated in Fig. 3. Multiple agents intend to move the box to the goal "G". Given that the canal becomes narrower, the agents must rotate the box to horizontal as it gets closer to the entrance of the narrowing part. Furthermore, there can be an obstacle "obs" on the way.

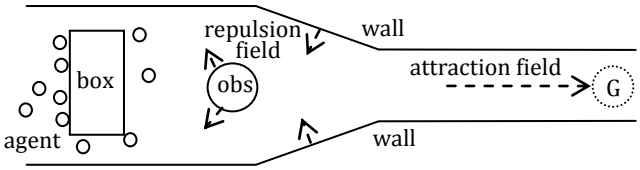


Fig. 3 Box-moving task used in case studies

The specific tasks can be expresses as follows.

T1 = <Orient><Goal>
T2 = <Move><Box>to<Goal>
T3 = <Rotate><Box>
T4= <Move> <Box>away from<Wall>
T5= <Sense><Social Field>

The task fields include the attraction field from the "goal" and the repulsion fields from the walls as well as the obstacle if present, as indicated in Fig. 3. For the "goal" field, a gravity-like field is applied, and for the "walls" and the "obstacle", a gradient based repulsion distribution is introduced to provide "warnings" of collision as agents get closer to them. The gradient distribution of the constraints (i.e., walls and obs) together with the sensory range of agents determine how much in advance agents can predict the collision and find ways to avoid it. In this simulation study, *higher positions* (i.e., *higher value*) in the field are more desirable to agents. For moving the box, an agent always tries to find a "low field position" around the box and from there to move the box toward a "high field position" which is often, but not always, the "goal" position.

We calculated the *object complexity* for the box which is the main object. The characteristics of the box include its dimensions *width* and *length* and its *orientation* angle. For simplicity, we consider the angle to be 90 degrees. Thus the objective complexity sums up to 2 for this item. There is one reciprocal (i.e., move and rotate) and two sequential activities (i.e., direct→move, and direct→rotate) that are interacting with each other. Therefore, the coordination complexity consists of three interconnected actions resulting in having the complexity of 3 for this portion leading to the total complexity of 12 for "with wall" situation, as indicated in Table1. The based on the similar calculation, the "open space" and "wall+obs" situations have complexity value of 7 and 13, respectively, shown in Table 1.

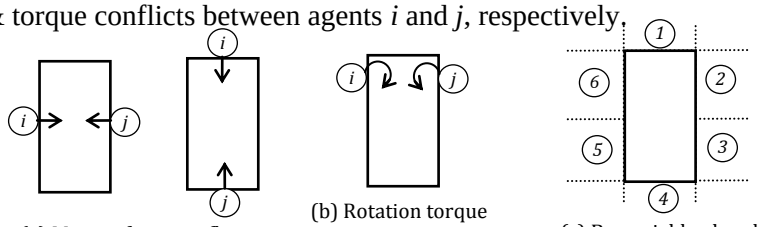
Table 1 Complexity measures of various box-moving situations

Situation	0: Open space	1: With Wall	2: With Wall + Obs
Complexity	7	12	13

The system is composed of *n* agents: $A = \{a_i\}$ $(i = 1, \dots, n)$. The initial positions of agents are randomly assigned but are always on the left side of the box. Guided by the task-field of attraction and repulsion, each agent contributes to the correct movement of the box in a way that the emergent movement of the box is toward the goal. Although this strategy (i.e., "non-social") works well for "open-space with a few obstacles" [4] when more constraints, such as "wall" and more "obs", are added, new strategies (e.g., "social structuring") are needed.

Social Rules

As mentioned above, social rules usually are designed to allow agents to avoid conflicts and/or to promote cooperation. In this case study, the social rules are set to provide guidance for agents to become aware of, and subsequently avoid, potential conflicts. Fig. 4 (a) and (b) illustrate possible force & torque conflicts between agents *i* and *j*, respectively.



(a) Moving force conflict (b) Rotation torque conflict (c) Box neighborhood

Fig. 4 Possible conflicts of agents *i* & *j;* and box neighborhood

To facilitate description of rules, we introduce the "box neighborhood" by defining 6 zones, as indicated in Fig. 4(c). Agents are aware of their location, i.e., their zone. Furthermore, they can broadcast their location and field density value to neighbor agents. The communication rule follows:

Social rule 1 (communication rule): <condition: enter box neighborhood> <recommended action: broadcast [location] and [field strength]>

When an agent receives broadcast from an agent in the neighborhood, it will attempt to determine if a force conflict or a torque conflict exists and then decide if it will take the recommended actions provided by the following conflicting avoidance rules:

Social rule 2 (force conflict rule): <condition: force conflict> <forbidden action: push in opposite-direction in opposite zone> <recommended action: find a new location>

Social rule 3 (torque conflict rule): <condition: torque conflict > <forbidden action: push in opposite-direction in opposite zone> <recommended action: move to next neighbor zone>

Agents have the option to ignore any or all of the above three rules. When the probability for agents to follow the rules decreases, we say that the system is less socially active, and otherwise more socially active.

Results

Fig. 5 illustrates a series of screenshots of a typical simulation run. The large box appeared to be a collection of small boxes because of an implementation difficulty. It should be considered as a single large box.

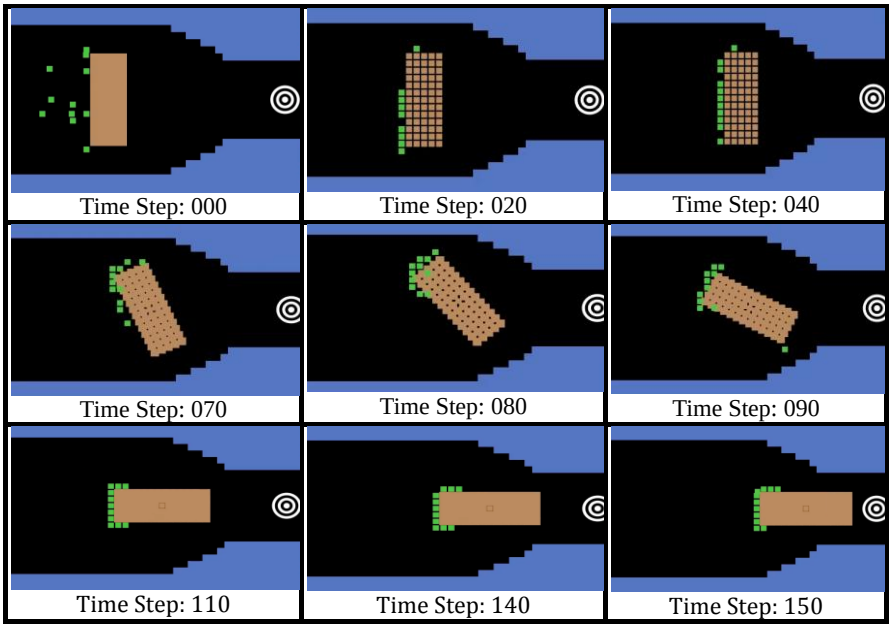


Fig. 5 Screenshots of a typical simulation run

Fig. 6 illustrates the comparison of success rate results for social (SRBR) and non-social (FBR) strategies for the "with wall" situation with varying number of agents. All results indicated in the graphs are averages of 100 simulation-runs for that specific setting. For non-social strategy (i.e., no social rule & no structuring), the success rate for 10-agent case is 0; no simulation runs could complete the task. Adding more agents increases the overall system complexity, resulting in better success rate. It can be seen that for this specific case study, 11-agent and 13-agent appear to be *critical* numbers by which the success rate jumps. The social structuring approach proves to be more reliable. The success rate remains 100% for all agent number settings. The increase of system complexity from adding social rules, and consequently social structures, has made the system more effective to deal with complex tasks.

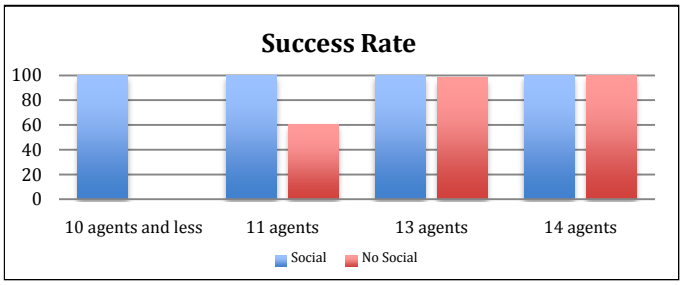


Fig. 6 Success rate comparison for social (SRBR) and non-social (FBR) strategies for the "with wall" situation with varying number of agents

Fig. 7 shows the comparison of total effort and time duration for completed (i.e., successful) simulation runs for non-social and social strategies. For non-social strategy, because no simulation run was successful for 10-agent or less case, there was no data for comparison. It is interesting to see that for the 11-agent case, the absence of social rules and structuring has made the system more efficient. This means that for the 60% completed runs (see Fig. 6), no social rule is more efficient than having social rules. This is because social structuring incurs "over-head" in task processing. However, the cost for this added efficiency is the 40% failed runs.

For cases where agent number is larger than 11, the social and non-social have the comparable success rates (see Fig. 6), but the social structuring strategy appears to be more efficient in terms of both effort and time duration. The implication of these results is important: while increasing complexity from (b) to (c) in Fig. 1 can be realized by either social structuring or adding more agents, the "impact" of them is different. Adding not-enough agent-power may run risk of failures and adding too much agent-power may lead to waste of time and effort. On the other hand, adding proper social structuring may remove the failure risk and maintain an adequate level of efficiency.

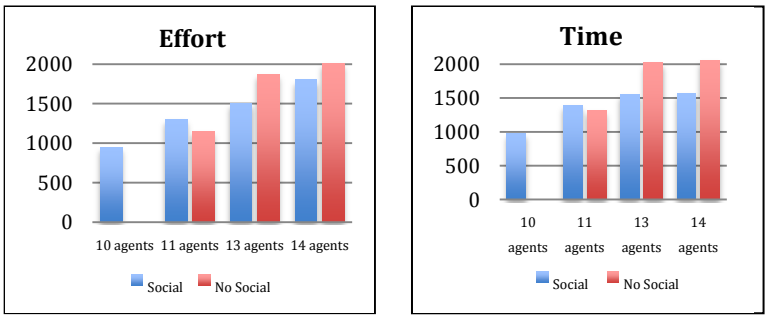


Fig. 7 Effort and duration time comparison for social (SRBR) and non-social (FBR) strategies for the "with wall" situation with varying number of agents

The change of social complexity over simulation time in a typical simulation run with social structuring strategy and 12 agents are shown in Fig. 8. As shown in the figure, social complexity increases when agents start to communicate with each other by following *social rule 1* and "help" each other by following social *2 & 3* when rotating the box in the middle of the process. Social complexity through social structuring varies over time; it increases when needed by the task situation (rotate the box) and decreases when the situation is resolved. This task driven variability is the key difference from the agent complexity obtained through adding more agent-power. While adding more agents somehow relies on "randomness" to increase the success rate and consequently looses efficiency, social rule based self-organization builds competence through local, bottom-up but explicit structuring efforts.

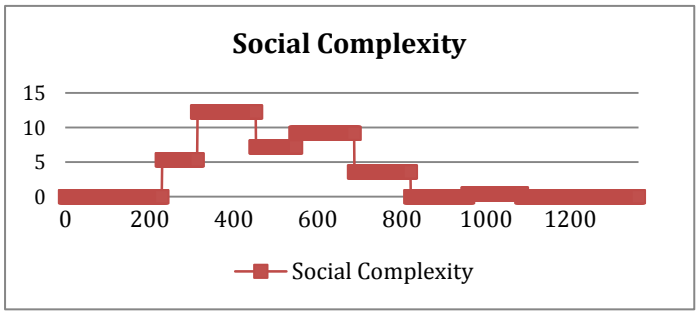


Fig. 8 Social complexity during the process of moving box towards goal with SRBR strategy and 12 agents

To further explore how more complex tasks demand social structuring, we carried out simulations for the "with wall+obs" situation. The task complexity measure for this situation is 13 (see table 1), more complex than the "with wall" situation. For this situation we only explored the cases for 14 to 18 agents. Fig. 9 shows the success rate comparison of two strategies and Fig. 10 the comparison of effort and time duration.

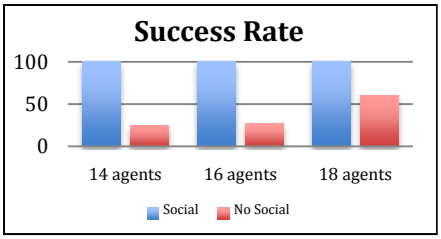


Fig. 9 Success rate comparison for social (SRBR) and non-social (FBR) strategies for the "with wall+obs" situation with varying number of agents

It can be seen from Fig. 9 that the success rate for non-social strategy decreased dramatically even with more agents (see Fig. 6). However, the social rule based structuring strategy remains to be 100% successful.

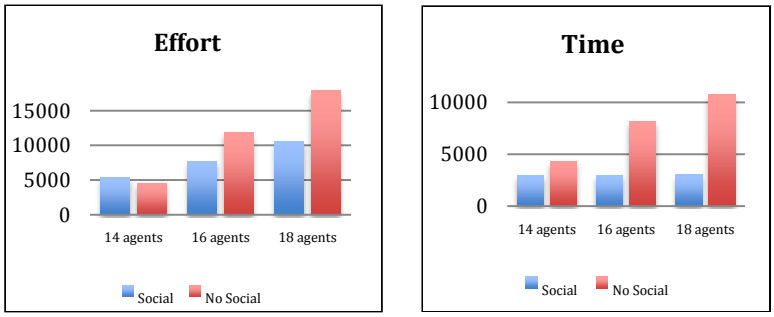


Fig. 10 Effort and time duration comparison for social (SRBR) and non-social (FBR) strategies for the "with wall+obs" situation with varying number of agents

For effort comparison, the non-social strategy is again more efficient than the social one for the 14-agent case, as shown in Fig. 10. However, its success rate is only 23% (see Fig. 9). Overall, the efficiency for non-social strategy is much worse than that for the social strategy. By comparing Fig. 10 with Fig. 8, it can be seen that the more complex task "with wall+obs" is more in need for emergent structural complexity of the system. However, when more agents are added into the already social-rule based system, there is only increase of effort and no improvement of time duration, as shown in Fig. 10. From the above results, it can be seen that devising proper social rules and adequate number of agents is important for designing CSO systems.

Concluding Remarks

As domain tasks become more complex, the engineered systems become more complex by moving from rigid and tightly organized formations into those of more components and more interactions. A potential issue with this top-down or ordered-to-disorder approach is the unintended and unknown interactions that may cause failure of the whole system. An alternative approach is to start with simple and disorganized agents and then move bottom-up and disordered-to-ordered by devising dynamic structures through self-organization. In this research, we explored the sources of task complexity by defining various complexity types and investigated how social rule based behavior regulation can be applied to allow dynamic structures, hence system complexity, emerge from self-

interested agents. The case study results have demonstrated the potential of effectiveness of our proposed approach and shed some useful insights.

- Increasing complexity from disorder can be achieved through adding more agents or devising structures. However, the former only has limited effect. When tasks become more complex, adding agents can hardly reach 100% success rate and the efficiency for the successful runs is low. On the other hand, devising dynamic structures can make the system more adaptable. Not only the success rate is always 100% but the efficiency is well maintained with changing task complexity (from "with wall" to "with wall+ obs") and varying number of agents (from 8 to 18). This result is consistent with Huberman and Hogg's [15] conjecture that higher structural complexity makes system more adaptable.
- When a relatively disordered system can complete a task by a certain probability, for this completed task, its efficiency can be better than structured systems; and this happens only for a small window of number of agents. The reason behind can be that dynamic structuring incurs over-head. However, the efficiency gain of the disorderliness is based on the high risk of failures.
- There can be tipping points of matching between the task complexity and system complexity. Adding one more agent from 10 to 11 can increase the success rate from 0 to 60% (Fig. 6), from 16 to 18 causes change from 25% to 60% (Fig. 9). This tipping point phenomenon can be due to the lack of social structuring of the system or it may be a result of mismatch between the highly complex task and not-so-complex system. Future work is needed to understand the real causal relations.

Our ongoing work explores the properties of various types of task complexity and their demands for corresponding types of structural complexity of the CSO system. Along the way, we will include more close-to-real engineering tasks and gradually make our CSO systems more real and practically functional.

This paper is based on the work supported in part by the National Science Foundation under Grants No. CMMI-0943997 and No. CMMI-1201107. 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 National Science Foundation.

References

1. Ashby WR (1956) An introduction to cybernetics. Taylor & Francis.
2. Zouein G, Chen C, Jin Y (2010) Create Adaptive Systems through 'DNA' Guided Cellular Formation. Design Creativity 2010: 149.
3. Chiang W, Jin Y (2011) Toward a meta-model of behavioral interaction for designing complex adaptive systems." IDETC2011-48821, Aug.29-31.
4. Chen C, Jin Y (2011) A behavior based approach to cellular self-organizing sys. design. IDETC2011-48833, Aug.29-31, Washington, DC
5. Simon HA (1962) The architecture of complexity. Proceedings of the American Philosophical Society: 467–482.
6. Williams EL (1981) Thermodynamics and the Development of Order. Creation Research Society Books.
7. Ferguson S, Lewis K 2006, Effective Development of Reconfigurable Systems Using Linear State-Feedback Control, AIAA J.,44/4, pp.868-878.
8. Martin MV, Ishii K (2002) Design for variety: developing standardized & modularized product platform architectures. Res. in Eng. Design 13/4,213-235.
9. Unsal C, Kilic H, Khosla P (2001). A modular self-reconfigurable bipartite robotic system: implementation and motion planning, Kluwer Autonom. Robots 10, pp.23-40.
10. Yim M , Zhang Y. Duff D (2002), Modular Robots, IEEE Spectrum 39/2, pp.30-34 .
11. Shen WM, Krivokon M, Chiu H, Everist J, Rubenstein M, Venkatesh J (2006). Multimode locomotion for reconfigurable robots. Autonomous Robots, 20(2), pp.165–177.
12. Horling B , Lesser V (2004) A Survey of Multi-Agent Organizational Paradigms. The Knowledge Engineering Review 19 (4): 281–316.
13. Galbraith JR (1977) Organization Design. Addison-Wesley Reading, MA.
14. Brooks CH, Durfee EH (2003) Congregation formation in multi-agent systems."Autonomous Agents and Multi-Agent Systems 7 (1): 145–170.
15. Huberman B, Hogg T (1986) Complexity and adaptation, in Physica D, Vol. 22/3, pp.376-384.
16. Wood RE (1986) Task Complexity: Definition of the Construct. Organizational Behavior and Human Decision Processes 37(1):60–82.
17. Campbell DJ (1988) Task complexity: A review and analysis." Academy of Management Review 13 (1): 40–52.
18. Gell-Mann M (1995). What is complexity? Complexity 1(1): 16-19.
19. Bonchev, D. 2004. Complexity analysis of yeast proteome network. Chemistry & Biodiversity 1 (2): 312–326.
20. Wilensky U (2001). Modeling nature's emergent patterns with multi-agent languages. Paper presented at EuroLogo 2001. Linz, Austria.