Causal Coupled Mechanisms: A Control Method with Cooperation and Competition for Complex System

1st Xuehui Yu
Faculty of Computing
Harbin Institute of Technology
Harbin, China
yuxuehui0302@gmail.com

2nd Yi Guan
Faculty of Computing
Harbin Institute of Technology
Harbin, China
guanyi@hit.edu.cn

3rd Xinmiao Yu
Faculty of Computing
Harbin Institute of Technology
Harbin, China
1190201203@stu.hit.edu.cn

4th Xue Li
Faculty of Computing
Harbin Institute of Technology
Harbin, China
20s103245@stu.hit.edu.cn

5th Jingchi Jiang*
the Artificial Intelligence Institute
Harbin Institute of Technology
Harbin, China
jiangjingchi@hit.edu.cn

Abstract—Complex systems are ubiquitous in the real world and tend to have complicated and poorly understood dynamics. For their control issues, the challenge is to guarantee accuracy, robustness, and generalization in such bloated and troubled environments. Fortunately, a complex system can be divided into multiple modular structures that human cognition appears to exploit. Inspired by this cognition, a novel control method, Causal Coupled Mechanisms (CCMs), is proposed that explores the cooperation in division and competition in combination. Our method employs the theory of hierarchical reinforcement learning (HRL), in which 1) the high-level policy with competitive awareness divides the whole complex system into multiple functional mechanisms, and 2) the low-level policy finishes the control task of each mechanism. Specifically for cooperation, a cascade control module helps the series operation of CCMs, and a forward coupled reasoning module is used to recover the coupling information lost in the division process. On both synthetic systems and a real-world biological regulatory system, the CCM method achieves robust and state-of-the-art control results even with unpredictable random noise. Moreover, generalization results show that reusing prepared specialized CCMs helps to perform well in environments with different confounders and dynamics.

Index Terms—complex system control, causal reasoning, hierarchical reinforcement learning

I. INTRODUCTION

Control methods for complex systems are of critical importance [1]. These complex systems represented by biological systems, transportation systems, and robotic systems have some common properties, such as the difficulty of understanding their dynamics and emergencies, which lead to a series of challenges to control them in our desired way [2]. By analysing many practical control tasks, we sum up three major challenges as follows:

* C1: Computational complex dynamics, sometimes even with emergent properties and notably delays [3], [4];
* C2: Unpredictable random external noises;
* C3: Heterogeneity among different objects. For example, different patient groups have different responses to treatment, yet the desired outcome is the same.

To solve the above three challenges for industrial application, improving the robustness and generalization of the control method is the core.

One important hallmark of human cognition is to exploit the modular structures of complex systems, which can be divided into multiple general and independent units [5]. For example, the glucose-insulin control system can be divided into insulin subsystem, glucose subsystem and other unit process models, and each needs to inform and influence each other.

![Fig. 1. Modularization of a glucose-insulin control system. The glucose-insulin control system can be divided into insulin subsystem, glucose subsystem and other unit process models, and each needs to inform and influence each other.](image-url)
field of causal reasoning, known as "causal modules". In causal reasoning, the independence between causal modules is a prerequisite for performing the division, and localized control [8]. However, causal boundary detection based on independence is intractable for high-dimensional variables. To simplify the detection of causal boundaries, an automatic division algorithm to rationalize the control tasks of each mechanism is necessary. Once agents learn the skills of controlling each mechanism by localized interventions, they can also control the entire system by combining and reusing these atomic skills. Reusing learned skills can solve transfer problems and achieve a stronger generalization.

After automatically dividing a complex system into multiple causal modules, how to concatenate these sub-modules to recover the original characteristics of the complex system needs to be considered. As phased by Aristotle, "the whole is greater than the sum of its parts;" properties of complex systems are not a simple summation of their independent functional mechanisms. Since the existing methods (modularity study [5] and causal reasoning [9], [10]) assume that only sparse interactions exist between the divided independent mechanisms, there is little attention to the cooperation between modules. However, the sparse interaction is a strong assumption that violates the laws of natural biological systems. As shown in Fig. 1, the liver can finish the degradation of insulin independently; the insulin flux which gets into the liver depends on the apportionment of the total insulin flux between plasma and liver. Plasma and the liver need to work together to maintain the entire system’s stability. So agents must consider how to reuse the independent functional mechanisms and complete their information transfer in the second step.

In this paper, we propose a control method Causal Coupled Mechanisms (CCMs) with state-of-the-art robustness and generalization, in which mechanisms contain full competition and cooperation. Competition and cooperation come from "The Origin of Species", which helps each natural biological community to complete the allocation of natural resources. In our modelling of a complex system, resources are the values of observable variables. An agent observes variables of systems and establishes control methods to assign them to each CCM (i.e. automatic modularization). Competition means that when an entire system is modularized, each CCM competes for resources of the whole system to maintain their local functional integrity and maintain their steady state operation. In other words, competition makes each CCM have the appropriate number of variables. The cooperation process considers how to reorganize the independent functional mechanisms. A hierarchical reinforcement learning (HRL) architecture is inspired by hierarchical processing information of human cognition. The high-level policy divides the complex system into multiple competing CCMs. The low-level policy focuses on each CCM to maintain its regular operation. In cooperation processing, a cascade control module is used for hindsight and feedback regulation and a forward coupled reasoning (FCR) module recovers helpful coupling information between variables and reduces computational complexity.

We selected a real-world biological regulatory system, the glucose-insulin system [6], [7], to verify the robustness and generalization of CCM in control problems faced with the above three challenges. Generalization experiment is taken in environments with totally different confounders and dynamics. In addition, we design three validation experiments to verify the performance of CCM on the three challenges respectively. The results show that CCM results in a diminished and dispersed effect of uncertainties and improves generalization. Finally, the visualization results of modularity show that different mechanisms have different functions.

II. RELATED WORK

In recent years, many advanced methods have been used for the control of complex systems due to the decreasing cost of computation and sensors. Model predictive control has become a dominant control strategy in research on intelligent operation [11], [12]. However such linear control models may not be accurate enough since the dynamics can be far from linear [13]. Mainstream control method often use deep learning mainly in model-free end-to-end controller settings, such as control of cyber-physical systems [14], [15]. And much of the success relies heavily on a reinforcement setup where the optimal state-action relationship can be learned via a large number of samples. To our knowledge, there are no high-performance model-free reinforcement learning (RL) models on biological control systems.

Recurrent Independent Mechanisms (RIMs) [5] is an impetus for this paper. RIMs is an architecture which divides complex systems into nearly independent transition dynamics, which communicate only sparingly through the bottleneck of attention. Such sparse interactions can reduce the difficulty of learning since few interactions need to be considered at a time, reducing unnecessary interference when a subsystem is adapted. Inspired by RIMs, we aim to divide the whole complex system into multiple functional mechanisms that have local independence. The key differences are that our CCMs makes full communication between mechanisms to ensure that the system formed by reuse can restore the performance of the original system as much as possible, thereby improving the overall stability of the system.

![Fig. 2](image-url)

(a) A causal graph dynamic. (b) The visualization of modularity.

We selected a real-world biological regulatory system, the glucose-insulin system [6], [7], to verify the robustness and generalization of CCM in control problems faced with the above three challenges. Generalization experiment is taken in environments with totally different confounders and dynamics. In addition, we design three validation experiments to verify the performance of CCM on the three challenges respectively. The results show that CCMs results in a diminished and dispersed effect of uncertainties and improves generalization. Finally, the visualization results of modularity show that different mechanisms have different functions.

III. BACKGROUND

This section starts with the introduction to the environment: 1) the first is the modelling approach called "causal graph dy-
Competition: Firstly, the high-level policy $\pi_\theta$ divides the entire system into a subsystem $CCM$; Then the low-level policy $\pi_\phi$ finishes the control of the $CCM$. These two steps are under the HRL framework. Cooperation: Time adjacent $CCMs$ need to contact under the help of two modules: a cascade control module helps the series operation of $CCMs$, and a forward coupled reasoning module is used to recover the coupling information lost in the division process.

**IV. CAUSAL COUPLED MECHANISMS**

The complex dynamical system can be divided into $K$ small subsystems (or mechanisms), i.e. an entire CGD is divided into $K$ small CGDs. We call such an independent and fully functional mechanism $CCM$ which has distinct functions that are unknown but can be sampled by taking actions. We propose the Coupled Mechanisms assumption: the causal generative process of a system’s variables is composed of semi-autonomous modules that need to inform and influence each other. The significant difficulty is modularizing automatically.

The HRL architecture is used for automatic modularization of the whole system. Fig. 3 presents the process of $CCM$ for the system in Fig. 2 (a). The high-level policy $\pi_\theta$ with competitive awareness divides the whole complex system into multiple functional modules. Specifically, $\pi_\theta$ gives a high-level action $a^h$ according to the observation $s^h$ of the whole system to separate a $CCM$. The $CCM$ will be used as the environment of the low-level policy $\pi_\phi$. There are two works for low-level policy: 1) controlling the subsystem $CCM$, and 2) helping coordinate between $CCMs$. Another two modules are needed for the cooperation, which is cascade control and FCR modules (more details can be found in Section IV-B).

**A. Competition between $CCMs$**

In the HRL framework, the high-level policy $\pi_\theta$ divides the whole system into multiple $CCMs$, in which competitive relationships should remain among $CCMs$. Firstly, $CCMs$ should have functional integrity, i.e. $CCMs$ should not be too small. Secondly, $CCMs$ should be independent, i.e. $CCMs$ should decouple with useless vertexes and avoid $CCMs$ too big to operate independently even single...
CCM is dominant. The low-level policy $\pi_\varphi$ finish the control task on each CCMs.

1) **High-level Policy for generating CCMs**: The high-level policy $\pi_\varphi$ instructs the low-level policy $\pi_\varphi$ via high-level actions $a^h$, which it samples a new every C steps. The high-level policy $\pi_\varphi$ optimises itself by evaluating the task effect and obtaining timely feedback from the low-level policy $\pi_\varphi$. The feedback is a single-step average reward $R^L$ that the low-level policy $\pi_\varphi$ gains at $C$ steps. The high-level RL is formalized with the quadruple $(S^h, A^h, P^h, R^h)$, whose elements are elaborated below.

**Actions.** Actions given by the high-level policy $\pi_\varphi$ are vertex sets, which are used for the whole system modularization. The effective idea is to minimize the loss of information in the modular process. We limit the action $a^h$ of the high-level policy $\pi_\varphi$ to a minimum vertex cut set between modifiable variables and target variables for this purpose. As shown in Fig. 2 (a), the minimum vertex cut set $\{B, C\}$ is chosen for modularization. Action space $A^h$ is the set of the minimum vertex cut set of $A \to F$.

**States.** The state $s^h_t$ is the description of the system and its statistical properties. More specifically, the $i$th dimension $s^h_{t,i}$ describes the statistical property of the $i$th minimum vertex cut set of $A \to F$:

$$s^h_{t,i} = \{\text{isCon}, \text{dis}, \text{num}, \ldots\},$$

where $\text{isCon}$ signifies whether the $i$th minimum vertex cut set is controllable (i.e. contained in the controllable subsystem), which is 1 if it is controllable, otherwise it is 0. $\text{dis}$ expresses the distance between the $i$th minimum vertex cut set and the target variables; $\text{num}$ shows the number of vertexes in the $i$th set. Beyond these, $s^h_{t,i}$ can also contain some other statistical properties, such as in-degree and out-degree.

**Transition.** The transition $P^h(s^h_{t+1}, a^h_t, s^h_t)$ is the state transition probability used to identify the probability distribution of the next state $s^h_{t+1}$, which is defined as a map function: $P^h : S^h \times A^h \to S^h$.

**Reward.** Three principles are presented to design the reward function: (1) The feedback information given by the corresponding low-level policy $\pi_\varphi$ of the current CCM $t_i$, e.g. $R^L_t = \max_{\pi_\varphi} \mathbb{E} \left[ \sum_{i=0}^{C} \gamma^i R^L(s^h_{t+i})/C \right]$; (2) vertexes in CCM $t_i$ are all uncontrollable, i.e. no cross between the CCMs; (3) The episode length shouldn’t be too long, which can cause CCMs too small. The high-level reward function $R^L_t$ at time $t$ can be defined as:

$$R^L_t = \begin{cases} 0, & \text{if } \text{isCon} = 0 \\ \alpha R^L_t + m - n, & \text{if } \text{isCon} = 1 \end{cases},$$

where $m$ and $n$ are positive real numbers and penalty items for offending against the principle (2) and (3) separately.

Finally, high-level action $a^h_t$ is used to generate CCM $t_i$ at time step $t$. Referring to the idea of cause-effect reasoning, the links from set $a^h_t$ to its parents and from set $a^h_{t-C}$ to its children are removed. The remained subgraph is current CCM.

2) **Low-level Policy for controlling CCMs**: The low-level policy finishes the control task of a CCM in $C$ steps and returns a single-step average reward $R^L$ for each control goal $g_c$ generated in the cooperation with other CCMs, which will detail in Section IV-B. In RL architecture, low-level RL is formalized with the quadartuple $(S^l, A^l, P^l, R^l)$, whose elements are defined as:

**Actions.** Agent’s action is to intervene modifiable variables of CCMs to $a^l_t$ at time step $t$. In causal theory, intervention vertexes are the "do operator". Low-level action space is the entire real space.

**States.** The low-level state $s^l_t$ is the values of vertexes in CCM $t_i$.

**Reward.** The mission objective is to maintain target variables in the range of goal $g_c$. The piece-wise reward function refers to the tip of entity-to-box distance [18] that uses an empirical approach aimed at maximising the ratio within goal $g_c$. So the low-level reward function is written as:

$$R^l_t = \omega - \text{dist}_{outside}(s^l_t; \mathbf{q}) - v \cdot \text{dist}_{inside}(s^l_t; \mathbf{q}),$$

where $\omega$ represents a fixed scalar margin, $0 < v < 1$ is a fixed scalar, $\mathbf{q} = [q_{\text{min}}, q_{\text{max}}] = [g_t - \epsilon, g_t + \epsilon] \in \mathbb{R}^{2d}$ is a query box; $\text{dist}_{outside}(s^l_t; \mathbf{q}) = \max(q(s^l_t) - q_{\text{min}}, 0) + \max(q_{\text{max}} - q(s^l_t), 0)$ is the central point of $q_{\text{max}}$ and $q_{\text{min}}$. If the BG level for the next state is in the target range, the agent will receive a positive reward. Otherwise, it will receive a negative reward.

The optimisal low-level policy obtains maximal single-step average reward $R^L_t = \max_{\pi_\varphi} \mathbb{E} \left[ \sum_{i=0}^{C} \gamma^i R^L(s^l_{t+i})/C \right]$ and backwards it to the high-level policy.

B. Cooperation between CCMs

Only with full cooperation among CCMs the whole system can secure stability. To adaptively schedule tasks and cooperate, a cascade control module and a FCR module are proposed.

1) **Cascade Control Module** : The agent’s goal is to complete the system’s stability, which means that the target variable of CGD is within the goal. Since the whole system is divided into multiple subsystems, the sub-goals of multiple stages are a phased decomposition of the init goal. The challenging initial goal will eventually be achieved by guiding agents to achieve periodic goals gradually. Hindsight experience replay [19] has given us great inspiration for generating sub-goals. The regenerated goals are generated by a function $m : S \to \mathcal{G}$, i.e. corresponding goal can be found for any state. Generating appropriate goals is a big challenge in multi-goal RL but is natural in the setting of our problem. That is because we cascade CCMs with their neighbours in space and time, shown on the right-hand side of Fig. 3. The goal of the previous stage can be sampled from the state of the later stage. For example, CCM $2$ is cascaded with CCM $1$, in which target variables $B, C$ are the modifiable variables in CCM $1$. The goal of target variables $B, C$ can sample from CCM $1$, i.e. sample from the optimised low-level policy $g_c \sim \pi_\varphi(g_c|\text{the state of CCM}_1, g_0)$. 

2559
More formally, low-level policy generates low-level action $a_{t+i}^l (1 \leq i \leq C)$ according to the state $s_{t+i-1}^l$ of $CCM_i$, i.e. $a_{t+i}^l \sim \pi_\phi(s_{t+i-1}^l, g_{t+i-1})$, where $g_{t+i-1}$ is sampled from $\pi_\phi(g_{t+i-1})$ (the state of $CCM_1$; $g_{t+C-1}$), $g_i (i \leq C)$ is the init goal used in $CCM_1$, moreover, the goal of the whole complex system.

2) Forward Coupled Reasoning Module: Some links between vertex cut set and its parent vertexes are removed during the generation of $CCMs$. When the vertex cut set contains more than one vertexes (i.e. modifiable variables), two issues arise 1) coupled information between modifiable variables will be lost, influencing action making; 2) agents need to control multiple actions simultaneously. Several studies that will be lost, influencing action making; 2) agents need to control multiple actions simultaneously. Several studies that are apparently related to the second issue, but are actually computational complex, including multi-agent [20] and multi-head [21] RL. We applied a global encoder for the two issues (shown in the middle part of Fig. 3). The global encoder learns the coupling relationship between modifiable variables and reconstructs the other multiple actions according to one action generated by the policy. More specifically, low-level action $a_{t+i}^l$ is the intervention value for the first vertex in $a_{t+i}^h$ at time step $t+i$. The $j$th intervention value $v_{t+i}^{(j)}$ is defined as:

$$v_{t+i}^{(j)} = \begin{cases} a_{t+i}^l, & \text{if } j = 0 \\ \text{RNN} \left( v_{t+i}^{(0)} - v_{t+i-1}^{(j)} \right), & \text{if } j \neq 0 \end{cases},$$

where $0 \leq j \leq |a_{t+i}^h|$, $|a_{t+i}^h|$ is number of vertexes in $a_{t+i}^h$, $0 \leq i \leq C$. Low-level action is $v_{t+i}^{(0)} = a_{t+i}^l \sim \pi_\phi(s_{t+i-1}^l, g_{t+i-1})$. $\text{RNN}_\xi$ is an artificial neural network with parameter $\xi$, i.e. recurrent neural network. It takes the history values in and generates the current $v_{t+i}^{(j)}$.

C. Optimization and Training

In this section, we discuss how to optimize our framework. The objective function of the low-level policy network is to maximize the expectation of accumulated rewards of hierarchical decisions,

$$J_L(\theta) = \mathbb{E} \sum_{i=0}^C \gamma^i R_{t+i}^L / C,$$

where $C$ is the maximal low-level episode length. The objective function of the high-level policy is,

$$J_H(\phi) = \mathbb{E} \sum_{t=0}^T \gamma^t R_{t+i}^H / T,$$

where $T$ is the maximal high-level episode length. Besides, we use policy gradient methods [22] to optimize both high-level and low-level policies, and the particularized loss function varies by different RL methods in the experiments. The objective function for the FCR module is cross-entropy,

$$J_{FCR}(\xi) = -\sum_{i=1}^N \left( \tilde{v}_i ln(v_i) + (\bar{v}_i - 1)ln(1 - v_i) \right),$$

where $v_i$ is the output of the global encoder $\text{RNN}_\xi$, $\bar{v}_i$ is the truth value from environment, and $N$ is the batch size.

Fig. 4. Experiment 1: (a) Average reward earned by the high-level agents in two environments. (b) The top left of the figure is causal graph for $Env_1$. A pointed arrow represents a linear relationship, and $w$ on the pointed arrow represents a constant. The down left of the figure is the causal graph for $Env_2$. We use a dotted arrow to represent a functional relationship, either activation or repression, with an implied explicit delay. The independent functional $CCMs$ for $Env_1$ and $Env_2$ are in the top right-hand and bottom right-hand separately.

V. EXPERIMENTS

The main goal of our experiments is to show that the use of $CCMs$ improves robustness and generalization across changing environments and in modular tasks and to explore how it does so. We set up experiments for the three challenges mentioned above respectively. Experiment in Section V-A corresponds to complex computations of challenge $C1$; Section V-B corresponds to random external noise of $C2$; Section V-D carries out generalisation verification which verifies $CCMs$ could deal with $C3$. Besides, an ordinary differential equation system [6], [7], which contains three challenges, is applied for robustness and module functional verification in Section V-C and then used for generalisation verification in Section V-D.

In $CCMs$, we use the Advantage Actor-Critic (A2C) [23] to optimize both high-level and low-level policies and a Multi-layer perception (Mlp) to produce the policy, "CCM A2C-MlpPolicy" for short. As the baseline, we use several high-performance non-hierarchical RL algorithms: A2C-MlpPolicy (an A2C with Mlp policy); A2C-LstmPolicy (an A2C with long short-term memory policy); PPO-MlpPolicy (an proximal policy optimization [24] with Mlp policy).

A. Experiment 1: Computational complex dynamics

We adopt two CGDs as environments and verify the impact of increased complexity (i.e. challenge $C1$). The maximum length of an episode is 100 for the two environments.
### TABLE I  THE SINGLE-STEP REWARDS FOR THREE EXPERIMENTS. ALL METHODS IN EXPERIMENTS 1 AND 2 HAVE LEARNT HOW TO CONTROL THE SYSTEMS WELL, BUT THE CCMs MODEL OBTAINED A HIGHER REWARD. IN THE REAL-WORLD SYSTEM (EXPERIMENT 3), THE BASELINES DEGRADED SO MUCH THAT THEY COULD NOT COMPLETE THE TASK, WHILE THE CCMs STILL OBTAINED A POSITIVE REWARD.

| Experiment 1 | A2C-MlpPolicy | A2C-LstmPolicy | PPO-MlpPolicy | CCM A2C-MlpPolicy (ours) |
|--------------|----------------|----------------|---------------|--------------------------|
| Simple       | 24.00          | 19.14          | 23.89         | 24.00                    |
| Direct Hill Regulation | 16.43          | 16.70          | 14.34         | 21.54                    |
|              |                |                |               | CCM1                     |
|              |                |                |               | CCM2                     |
|              |                |                |               | CCM3                     |
| Experiment 2 | A2C-MlpPolicy | A2C-LstmPolicy | PPO-MlpPolicy | CCM A2C-MlpPolicy (ours) |
| Simple       | 22.73          | 20.70          | 23.01         | 23.09                    |
| Direct Hill Regulation | 7.14           | 8.19           | 6.91          | 9.36                     |
| Big Direct Hill Regulation | 7.26           | 7.27           | 7.09          | 9.92                     |
|              |                |                |               | CCM1                     |
|              |                |                |               | CCM2                     |
|              |                |                |               | CCM3                     |
| Experiment 3 | A2C-LstmPolicy | PPO-MlpPolicy | CCM A2C-MlpPolicy (ours) |
| Insulin (No noise) | -899.93        | -1027.29       | 6.52          |
| Insulin (Random large noise) | -899.93        | -1027.29       | 6.52          |

**Env 1: Simple.** This CGD contains five variables. With [25] as reference, each node \( x_i \in \mathbb{R} \) is a Gaussian random variable. Parentless observed variables have distribution \( N(\mu = 0.0, \delta = 0.1) \). A node \( x_i \) with parents \( pa(x_i) \) has conditional distribution \( p(x_i|pa(x_i)) = N(\mu = \sum_j w_{ij}x_j, \delta = 0.1) \), where \( x_j \in pa(x_j) \) and \( w_{ij} \) is a constant. The graph structure is represented at the top left of Fig. 4 (b).

**Env 2: Direct Hill Regulation.** Physiological processes typically have emergent properties of many parameters or time delays between the initiation of the physiological mechanism and the resulting functional output [4], [26]. To model the phenomenology of many biological networks, we refer to explicit-delay modeling [4] to build CGDs. Specifically, the conditional distribution \( p(x_i|pa(x_i)) = N(\mu = \sum_j f_{ij}x_j, \delta = 0.1) \) is changed, where \( f_{ij} \) represents explicit-delay function (i.e. \( f_{ij} \) is a direct hill regulation function in Env 2). The other situations keep the same, as shown at the top right of Fig. 4 (b).

**Result.** Fig. 4 (a) shows the result of the high-level policy on modularity. The single-step average reward goes steadily up in the two environments, although it decreases with the increase of system complexity. From the visual results in Fig. 4 (b), the whole system is divided into two parts and the vertex cut set \{2\} is chosen. An obvious phenomenon can be found that the CCM2 in Fig. 4 (b) contains a collider \( x_1 \) (i.e. a nonendpoint vertex at where two arrowheads meet) which will increase complexity. The existence of collider \( x_1 \) is perhaps a good reason that agents choose to modularize in \( x_2 \). In the child table “Experiment 1” of table I, the column headed “CCM1” represents the single-step reward for controlling the target variables of the whole system. CCMs achieves the highest reward and is relatively robust to increasing the complexity of systems, whereas the baseline’s performance degraded more.

**B. Experiment 2: Unpredictable random external noises**

Parentless target variables will randomly become a larger value (increased more than ten times) to verify robustness. Apart from causal graphs in Experiment 2, a new environment Env 3 is designed. The maximum length of an episode is 100 for the three environments.

**Env 3: Big Direct hill regulation.** Env 3 adds some variables and a double path based on Env 2 to increase system complexity, which can better show the impact of external noises (as shown at the top of Fig. 5 (b)). The double paths can help check whether the FCR module can restore coupled information between modifiable variables.

**Result.** Random external noise reduces the reward for both high-level policy and low-level policy, compared with “Experiment 1”

The performance on Env 2 degrades more severely because of the addition of computational complexity and random external noise. When evaluating in the environment with novel unseen noise, CCMs doesn’t achieve perfect performance but strongly outperforms the baselines (shown as child table “Experiment 2” of I). In the visualization results (Fig. 5 (b), CCMs is consciously trying to divide the Env 3 into three parts to reduce complexity. The divisions for Env 1 and Env 2 is the same as Fig. 4 (b) in “Experiment 1”. This indicates that the agent will use different modularity strategies depending on the system’s complexity.

The modularization of CCMs results in a diminished and dispersed effect of the huge noise, enhancing the target information and suppressing irrelevant noise. As shown in Fig. 5 (b), \( x_0 \) containing noise is divided into CCM1 and has diminished effect on CCM2. The agent has learned how to eliminate the influence of unseen noise on the system’s stability in a smaller graph CCM2.

**C. Experiment 3: A real-world biological regulatory system**

We verify the performance of CCMs on a real-world biological regulatory system [6], [7], which is suitable for CCMs. The model assumes that the glucose and insulin subsystems are linked one to each other by the control of insulin on glucose utilization and endogenous production (the original gives more detailed divisions). We can compare the
modularization results of CCMs with the actual functional modules to investigate the functionality of automatic modularization. According to the definition of CGD, the graph structure is shown in Fig. 6 (e). The glucose-insulin model has 26 free parameters that vary in different individuals and will be used in generalization experiment. The glucose-insulin meal model provides thirty individuals, and Adult #004 is selected to interact with CCMs. Each step simulates one minute, and the maximum length of an episode is 1440, i.e., one natural day. In this experiment, noises are designed as two types, i.e. with and without noise. In the real world, the uncertainties (e.g. sensor error, irregular eating habits and so on) in glucose control of diabetics can be linked to the random external noise.

Result. The results of high-level policy are shown in Fig. 6. In the child table “Experiment 3” of table I, baselines failed to control the system under the two noise settings while CCMs achieved good performance. The loss curve of global encoders is displayed in Fig. 6 (b), and the red circle represents the phenomenon of radiant explosion. In addition, we make a deeper analysis:

1) Analysis for multipath selection and the effect of FCR module. At the second step, the high-level agent needs to choose a double vertexes set for modularization. In the experiment with noises, high-level agents only learnt the $x_{12} \rightarrow x_3$, $x_6 \rightarrow x_10$, $x_6 \rightarrow x_7$, $x_5 \rightarrow x_10$ (the value on arrow represents the single-step average reward). In the noiseless experiment, high-level agents found three vertex cut sets $\{x_6, x_7\}$, $\{x_6, x_9\}$ and $\{x_4, x_9\}$, and chose the best one finally. The FCR for these three sets also convergence as shown in Fig. 6 (b). In summary, 1) the agents weighed up all feasible control paths and chose the best, 2) the choice of control path is directly related to the effect of FCR and 3) the FCR can extract useful coupling information.

2) Accuracy of realistic tasks. We take testing for 1440 steps and calculate the percentage time in the glucose target range of [70, 180] mg/dL (TIR) [27]. In the training task adult #004, the TIR is 100%, which means that the policy given by CCMs is of practical significance and can be used for glucose control tasks.

3) Functionality of the mechanisms. The four mechanisms have different functions. For example, $x_3 \rightarrow x_{12}$ in CCM1 represents the dynamic relationship between plasma and interstitial fluid glucose.

D. Experiment 4: Heterogeneity among different objects

The functional mechanisms in CCMs have an ingenious connection with the concept of affordances from cognitive psychology [28]. An agent should be ready to adapt immediately, even in an environment of uncertainty, executing skills which are at least partially prepared. This suggests that agents should contextually process sensory information, building representations of potential actions that the environment currently affords. We investigate how agents use prepared specialised CCMs to improve generalisation between different environments, which have important variation factors. Adult #004 interacts with CCMs for training, and thirty individuals are used for testing.

Result. (1) Analysis for generalization. Similar to “Experiment 3”, we take testing for 1440 steps and calculate the TIR for 30 individuals, which is shown in the column headed
The glucose profile over 1440 steps is employed for demonstration purposes over a testing period. Among 1440 steps, we apply three random noise interventions to the individuals. Fig. 7 shows the change of glucose under the control of the optimised CCMs. The value of glucose is always within the normal range in the adult group. Hyperglycemia occurred in Adolescent #002, Child #001 and Child #008, suggesting the child group differs more significantly from the adult group. In summary, CCM has perfect generalisation in the same population and certain generalisation in different populations.

VI. CONCLUSION
In this paper, we propose a hierarchical reinforcement learning method to control complex systems. This method incorporates the high-level agent with competition awareness and the low-level agent with cooperation awareness to search for the optimal control policy. To improve the robustness and generalization of our control algorithm, Causal Coupled Mechanisms, the high-level policy employs multi-goal learning to divide an entire system into multiple atomic causal modules, which can be effectively transferred and reused in similar but different tasks. To relax the independent assumption between causal modules, the low-level policy adopts coupled reasoning to deal with cooperative control problems. We conduct experiments on both synthetic systems and a real-world biological regulatory system to verify the advantages of our model. Compared to the existing methods, our method achieves the state-of-the-art results in addressing the complexity, random external noise, and heterogeneity of complex systems.

VII. ACKNOWLEDGEMENT
This study was supported in part by grants from the National Key R&D Program of China [2021ZD0113302], National Natural Science Foundation of China [62006063], and the Heilongjiang Provincial Postdoctoral Science Foundation (CN) [LBH-Z20015].

REFERENCES
[1] Y. Chen, Y. Shi, and B. Zhang, “Optimal control via neural networks: A convex approach,” arXiv preprint arXiv:1805.11835, 2018.
[2] R. Baheti and H. Gill, “Cyber-physical systems,” The impact of control technology, vol. 12, no. 1, pp. 161–166, 2011.
[3] C. Gupta, J. M. López, W. Ott, K. Josić, and M. R. Bennett, “Transcriptional delay stabilizes bistable gene networks,” Physical review letters, vol. 111, no. 5, p. 058104, 2013.
[4] D. S. Glass, X. Jin, and I. H. Riedel-Kruse, “Nonlinear delay differential equations and their application to modeling biological network motifs,” Nature communications, vol. 12, no. 1, pp. 1–19, 2021.
[5] A. Goyal, A. Lamb, J. Hoffmann, S. Sodhani, S. Levine, Y. Bengio, and B. Schölkopf, “Recurrent independent mechanisms,” arXiv preprint arXiv:1909.10893, 2019.
[6] C. Dalla Man, R. A. Rizza, and C. Cobelli, “Meal simulation model of the glucose-insulin system,” IEEE Transactions on biomedical engineering, vol. 54, no. 10, pp. 1740–1749, 2007.
[7] C. Dalla Man, D. M. Raimondo, R. A. Rizza, and C. Cobelli, “Gim: Simulation software of meal glucose—insulin model,” 2007.
[8] J. Pearl et al., “Models, reasoning and inference,” Cambridge, UK: Cambridge University Press, vol. 19, p. 2, 2000.
[9] G. Parascandolo, N. Kilbertus, M. Rojas-Carulla, and B. Schölkopf, “Learning independent causal mechanisms,” in International Conference on Machine Learning. PMLR, 2018, pp. 4036–4044.
[10] K. Madan, N. R. Ke, A. Goyal, B. Schölkopf, and Y. Bengio, “Fast and slow learning of recurrent independent mechanisms,” arXiv preprint arXiv:2105.08770, 2021.
[11] G. Van Cutsem, M. Kayal, D. Blum, and M. Pritoni, Comparison of MPC formulations for building control under commercial time-of-use tariffs. IEEE, 2019.
[12] G. Bianchini, M. Casini, D. Pepe, A. Vicino, and G. G. Zanvettor, “An integrated mpc approach for demand-response heating and energy storage operation in smart buildings,” in 2017 IEEE 56th Annual Conference on Decision and Control (CDC). IEEE, 2017, pp. 3865–3870.
[13] P. H. Shaikh, N. B. M. Nor, P. Nallagowden, I. Elamvazuthi, and T. Ibrahim, “A review on optimized control systems for building energy and comfort management of smart sustainable buildings,” Renewable and Sustainable Energy Reviews, vol. 34, pp. 409–429, 2014.
[14] T. Wei, Y. Wang, and Q. Zhu, “Deep reinforcement learning for building hvac control,” in Proceedings of the 54th annual design automation conference 2017, 2017, pp. 1–6.
[15] D. O’Neill, M. Levorato, A. Goldsmith, and U. Mitra, “Residential demand response using reinforcement learning,” in 2010 First IEEE international conference on smart grid communications. IEEE, 2010, pp. 409–414.
[16] P. Arrighi and G. Dowek, “Causal graph dynamics,” in International Colloquium on Automata, Languages, and Programming. Springer, 2011, pp. 54–66.
[17] O. Nachum, S. S. Gu, H. Lee, and S. Levine, “Data-efficient hierarchical reinforcement learning,” Advances in neural information processing systems, vol. 31, 2018.
[18] H. Ren, W. Hu, and J. Leskovec, “Query2box: Reasoning over knowledge graphs in vector space using box embeddings,” arXiv preprint arXiv:2002.05969, 2020.
[19] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, “Hindsight experience replay,” Advances in neural information processing systems, vol. 30, 2017.
[20] K. Zhang, Z. Yang, and T. Başar, “Multi-agent reinforcement learning: A selective overview of theories and algorithms,” Handbook of Reinforcement Learning and Control, pp. 321–384, 2021.
[21] Y. Flet-Berliac and P. Preux, “Merl: Multi-head reinforcement learning,” arXiv preprint arXiv:1909.11939, 2019.
[22] O. Sutton, D. McAllester, S. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” Advances in neural information processing systems, vol. 12, 1999.
[23] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in International conference on machine learning. PMLR, 2016, pp. 1928–1937.
[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.
[25] I. Dasgupta, J. Wang, S. Chiappa, J. Mitrovic, P. Ortega, D. Raposo, E. Hughes, P. Battaglia, M. Botvinick, and Z. Kurfth-Nelson, “Causal reasoning from meta-reinforcement learning,” arXiv preprint arXiv:1901.08162, 2019.
[26] O. D’Huys, R. Vicente, T. Erneux, J. Danckaert, and I. Fischer, “Causal reasoning from meta-reinforcement learning,” arXiv preprint arXiv:1909.11939, 2019.
[27] P. Cisek and J. F. Kalaska, “Neural mechanisms for interacting with a world full of action choices,” Annual review of neuroscience, vol. 33, pp. 269–298, 2010.
[28] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in International conference on machine learning. PMLR, 2016, pp. 1928–1937.
[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.