Narrative Documentation for the MAB-SCD Model

1. Introduction

In the school education scenario, Teachers and students are the main actors in the process of teaching and learning implementation of school subject curricula. Correspondingly, the curriculum provides a structured framework for teacher teaching and student learning. The MAB-SCD (Multi-Agent-Based Student Cognitive Development) simulation model represents a sophisticated endeavor tailored to comprehensively encapsulate the dynamic intricacies of students' cognitive structures throughout school curriculum instruction. Firmly grounded in the BDI (Belief-Desire-Intention) cognitive architecture, MAB-SCD distinguishes itself by seamlessly incorporating theories of Cognitive Diagnostic Assessment.

The hallmark of this model lies in its distinctive capacity to dynamically simulate the evolutionary trajectory of students' cognitive structures over time. This simulation unfolds as a dynamic process, intricately mirroring the learning journey of students within the educational context. An integral aspect of its functionality is the meticulous accumulation of information essential for the completion of learning tasks or the attainment of specific educational goals.

2. Conceptual model of the MAB-SCD

2.1 Define System Boundaries

In the realm of secondary school subjects, the initiation of new learning primarily occurs through teacher-led lectures in the classroom. Additionally, students engage in independent learning to reinforce their mastery of acquired knowledge and review areas where further understanding is needed. This independent learning serves the overarching goal of consolidating subject knowledge. The process of making connections is predominantly evident within specific subjects. It involves linking core knowledge within a subject and establishing connections between the teaching sequences implemented by the teacher and the overall structure of the curriculum and associated tasks. This interconnectedness facilitates a more comprehensive

understanding of the subject matter and enhances the overall learning experience for students. Consequently, our study delves into these nuanced processes to gain insights into the dynamic nature of learning in secondary school subjects.

2.2 Identify and Define Agents

In the MAB-SCD framework, four distinct types of Agents are defined to comprehensively simulate the dynamic changes in students' cognitive structures and knowledge acquisition over time, particularly within a semester or an extended duration.

Table 1 presents the role and characteristics or static attributes and behaviors or desicion-making process of each type of agent.

Table 1. Definition of Agents in MAB-SCD

Agent	Role	Static Attributes	Behaviour
Curriculum Agent	Provides a structured framework for teachers and students, outlining the content, learning objectives, and assessment criteria for each subject.	C1: Curriculum core concepts C2: Attribute hierarchy C3: Learning progression level	B1: Generating the ideal mastery pattern
Teacher Agent	Instructs students in the acquisition and mastery of new knowledge in accordance with the teaching objectives and instructional schedules.	C4: Instructional sequence C5: Instructional schedule	B2: Choosing current lecture content B3: Teaching a new lesson
Student Agent	Actively participates in learning, acquiring knowledge and skills in the curriculum.	C6: Cognitive ability C7: Maximum knowledge points C8: Knowledge mastery C9: Attribute mastery pattern	B4: Learning new knowledge B5: Reviewing unmastered knowledge B6: Reviewing mastered knowledge
LTM Agent	As a crucial component of the human memory system responsible for the storage of information over an extended period.	C10: Forgetfulness time threshold for knowledge C11: Mastered knowledge attribute set C12: Relationship collection among mastered attributes	B7: Managing LTM (including adding, deleting and modifying conceptual attributes)

(1) Curriculum Agent

In accordance with the Cognitive Diagnostic Assessment (CDA) theory, we have conceptualized the Curriculum Agent. As depicted in Table 1, the static attributes of the Curriculum Agent primarily consist of three characteristics, denoted as C1~C3. Notably, C1 signifies the teaching content associated with a set of core concepts in the curriculum during the teaching stage of a particular subject. For instance, it may encapsulate the core knowledge concepts designated for instruction in the seventh-grade mathematics curriculum. The specific determination of curriculum content draws guidance from authoritative curriculum standards and real-world school teaching scenarios. Each element in C1 corresponds to the term Cognitive Attribute in

the CDA, which is used to denote the knowledge, skills, and strategies that students are expected to acquire in a given course that are covered by the subject at that stage (Leighton et al., 2004; Tatsuoka, 2013). The breadth of content contained in each element corresponds to the Attribute grain size terminology in CDA, and the specific settings should be adjusted according to the actual learning situation and the teacher's teaching strategies.

C2 reflects the hierarchical relationships between attributes. As the core concepts involved have a clear organisational structure, the hierarchical relationships between attributes in the cognitive developmental approach are defined as a specific delineation of the links between identified cognitive attributes.

C3 indicates the course's teaching or student learning objectives. Using curriculum standards and expert guidance, the alignment between academic achievement levels and knowledge attributes students should master serves as an evaluation criterion for assessing students' academic progress in the course.

The Curriculum Agent involves the behaviour B1, which functions to generate all the ideal attribute mastery patterns. This process is based on input lists of knowledge concept attributes and the hierarchical structure they form, represented by Adjacency and Reachability Matrices.

(2) Teacher Agent

Research in educational psychology suggests that the teacher plays a fundamental and active role in the teaching process since he has the responsibility of the rhythm, content and sequence of the lecture (Ormazábal,2021).

Hence, for the Teacher Agent in the MAB-SCD model, our focus lies on two static attributes, C4 and C5. C4 represents the subject matter taught by the teacher during the course, while C5 outlines the teacher's teaching schedule. Following the principles of Cognitive Diagnostic Assessment (CDA), cognitive attributes are derived from core concepts, and their operation is not entirely independent; they may exhibit a mental order, logical sequence, or hierarchical relationship. Consequently, the teacher's content and teaching activities should align with these requirements. In this context, C4 represents a topological sequence conforming to hierarchical

relationships between attributes. On the other hand, C5 delineates the teacher's lecture plan and content over a specific time period, measured in days. The lectures encompass pedagogical content corresponding to the attributes in C1, and their overall sequence is based on C4.

The behaviour of the Teacher Agent specifically consists of two dynamic decisions labeled B2 and B3. They represent the teacher's actions of selecting the content for the current lecture and delivering a new lesson, respectively.

(3) Student Agent

The static attributes of the Student Agent encompass four main components (C6~C9). C6, representing cognitive ability, is considered an individual characteristic linked to students' academic achievement(Rohde & Thompson，2007; Tikhomirova, Malykh & Malykh, 2020). In this context, cognitive ability refers to the capacity for cognitive processing and the intellectual foundation that students possess within the school program.

C7 signifies the maximum number of knowledge points a student can review, establishing an upper limit on the selection of knowledge attributes during each review. C8 reflects the student's evolving mastery of all knowledge concepts throughout the course of study. C9 encapsulates the student's pattern of mastering course knowledge, encompassing the acquired knowledge attributes and adhering to the hierarchical structure established between these attributes. Furthermore, the integration of C9 and C3 enables the determination of the student's learning trajectory and academic level.

The behaviors of the Student Agent encompass B4-B6, where B4 involves students learning new knowledge through the instructor's lecture setting; B5 is the re-learning of knowledge attributes that were not mastered in lectures through independent learning, and B6 entails reviewing mastered knowledge attributes for knowledge consolidation.

(4) LTM Agent

In order to accurately model the intricate process of students' knowledge acquisition, involving precise simulation of storage and forgetting events in the

human brain. This study integrates insights from the Long-Term Memory (LTM) framework proposed by Chiriacescuand his team(2013). This conceptual framework provides valuable insights into the structure of human long-term memory based on ULM, serving as a reference for modeling a student's ability to manage learned knowledge concepts within the course.

The LTM Agent encompasses three static attributes, denoted from C10 to C12. C10 sets the maximum duration for a knowledge attribute to undergo decay in long-term memory. In essence, if students opt not to revisit a knowledge attribute beyond the specified C10 value, it is presumed that the strength of their memory for that particular knowledge point will diminish over time.

C11 constitutes the set of knowledge attributes in the long-term memory, representing a dynamically changing subset of C1. It reflects which of the knowledge points taught in the course a specific student has presently mastered. C12, on the other hand, is a subset of C2 that records the relationships between the knowledge attributes mastered by a student.

The behavior of the LTM Agent, labeled as B7, encompasses management operations on long-term memory. Specifically, these operations include adding, modifying, and deleting concept attributes stored in the LTM.

2.3 Inter-Agent Relationships

Figure 1 presents the UML class diagram for MAB-SCD, providing a graphical representation of the attributes, behaviors, and relationships among the four types of Agents. This visual depiction succinctly captures the structural and dynamic elements of the system, showcasing key information about each agent's properties, functionalities, and interconnections.

In the specific context of Student and Teacher Agents, the examination of their behaviors and decision-making processes revolves around the BDI (Belief-Desire-Intention) architecture (Howden et al.,2001; Pokahr, Braubach & Lamersdorf, 2005; Kennedy, 2011). By incorporating the behaviors of other agents into the framework, it offers a structured perspective for analyzing and understanding how beliefs, desires, and intentions influence the dynamic behaviors of agents in a

simulated environment.

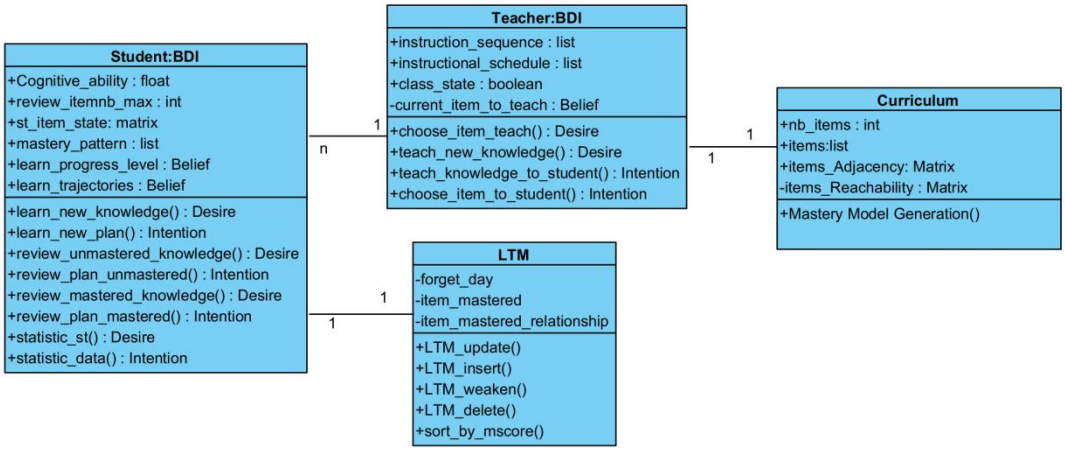

Figure 1. Class diagram of MAB-SCD

To enhance the elucidation of interaction and communication among diverse Agents in MAB-SCD, this study conducts a thorough analysis of the system's activity diagram. This examination aims to comprehensively understand and capture the dynamics of students' learning behaviors, their decision-making processes, and their interactions with the Teacher Agent and the Long-term Memory Agent within the school curriculum learning scenario.

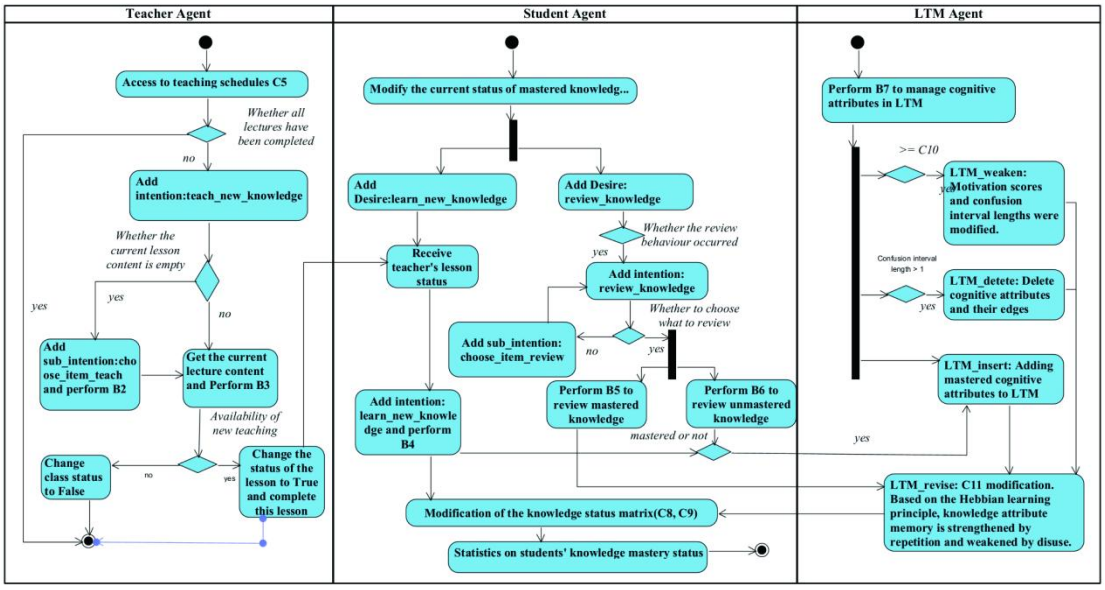

Figure 2. Activity diagram of MAB-SCD

Figure 2 provides a comprehensive visualization of the key activities undertaken by each type of Agent within a single simulation cycle in our MAB-SCD framework. It is important to note that, depending on the research problem's requirements,

multiple simulation cycles may be traversed. The figure also illustrates the communication dynamics between Agents that engage in interaction relationships.

Teacher Agent and Student Agent Interaction: The Teacher Agent and Student Agent interact to simulate classroom activities and instruction. During phase B3, the Teacher Agent communicates with the Student Agent to determine the current curriculum content and facilitates learning by providing new material. The Student Agent then engages in learning activities during phase B4.

Student Agent and LTM Agent Interaction: The interaction between the Student Agent and the LTM Agent within the MAB-SCD framework models the processes of knowledge storage and management. This interaction encompasses several key phases: During phases B4 and B6, the Student Agent acquires new cognitive attributes and stores them in the LTM Agent, thereby simulating the learning process. In phase B5, the Student Agent collaborates with the LTM Agent to update the state of cognitive attributes (C11), reflecting the consolidation of newly acquired knowledge. Additionally, according to the Hebbian learning principles (Caporale & Dan, 2008), memory for knowledge attributes is reinforced through repetition and weakened by disuse. Specifically, during phase B5, the Student Agent adjusts parameters such as the confusion interval length and motivation score for cognitive attributes (C11) in collaboration with the LTM Agent. This adjustment reflects the ongoing consolidation of knowledge. Conversely, if knowledge remains unused for an extended period—i.e., when the time since the last access of cognitive attributes in the LTM exceeds the threshold C10—knowledge decay occurs. If the confusion interval length surpasses 1, all subsequent nodes associated with the cognitive attribute are removed from the LTM, indicating knowledge forgetting.

It is important to note that, depending on the research problem's requirements, multiple simulation cycles may be traversed. The figure also illustrates the communication dynamics between Agents that engage in interaction relationships. Notably, the interaction between the Teacher Agent and the Student Agents is predominantly facilitated through the "Lecture" activity. When the Teacher Agent initiates the Lecture Desire intent, it formulates the lecture content based on the input

lecture plan, subsequently communicating it to the students. During this phase, all Student Agents execute the Learning New Knowledge intention.

This aims to simulate the storage and management of knowledge items within LTM through students' course learning and independent revision activities. This dual-process involves adding new content to the LTM, achieved by invoking the LTM_insert function in the intent of new lectures and in the intent of reinforcing the learning of previously acquired but not yet mastered knowledge items. Simultaneously, the model emphasizes the consolidation of learning for mastered knowledge items in the LTM, ensuring a firm grasp for review activities. Recognizing the characteristics of human long-term memory, wherein disuse can lead to memory attenuation or even forgetting, the simulation model dynamically reflects changes in the internal state of LTM items over time. This dynamic representation aligns with the real-world dynamics of long-term memory, enhancing the realism and effectiveness of the MAB-SCD framework.

3. Definition of the MAB-SCD

3.1 Time and Event Handing

Specifying how time is modelled is crucial for accurately capturing the nuanced dynamics of knowledge transfer and cognitive evolution within the context of a typical secondary school curriculum. The MAB-SCD simulation is configured with a time step of 1 day. The total simulation duration is continuous and measured in semesters, typically spanning 16 to 18 teaching weeks per semester.

Within the simulation framework, lecture activities, designed to impart new knowledge, are intricately tied to the teacher's schedule, aligning with the specific curriculum for each class on a day-to-day basis. This meticulous alignment mirrors the structure of the secondary school curriculum. Students' daily independent learning activities introduce stochastic elements, adding randomness to the simulation. Self-directed learning includes two stochastic events: re-learning previously taught but not fully mastered knowledge and repeating mastered knowledge. The focus is solely on whether the student engages in each event within a simulation time step, not on the event's specific duration within a day.

Additionally, each student agent stores the acquired knowledge in its LTM Agent, which is a simulation of an individual's relatively permanent store of knowledge. The key events involved in the LTM Agent include (i) the assimilation of newly acquired knowledge and skills into the LTM and (ii) the application of Hebbian learning principles(Caporale & Dan, 2008), wherein knowledge and skills within the LTM undergo strengthening through repetition and weakening through disuse. The temporal modeling of each student's LTM is meticulously executed in daily increments. This offering a detailed exploration of the persistent and dynamic nature of the students' Long-Term Memory processes.

3.2 Input

Analyzing the inputs of the MAB-SCD model enhances understanding of their dynamics and decision-making from the perspective of each Agent type.

The input information for the Curriculum Agent is comprised of a set of cognitive attributes that are systematically associated with the core concepts and skills of a subject course. Each element within this comprehensive list contains the identifier of the item and the difficulty coefficient corresponding to the knowledge point it represents. The hierarchical relationships between individual items are effectively captured through the use of adjacency and reachability matrices. These matrices serve as robust representations of the hierarchical structure within the course, offering insights into the interconnections and accessibility of various knowledge points. The learning progression levels within the matrices align with the specified course learning objectives, delineating a progression from lower to higher learning objectives. Algorithmically derived ideal mastery patterns, informed by the Reachability Matrix and hierarchical relationships between items, form the basis for the simulation. These ideal patterns are subsequently employed as inputs to determine the actual mastery patterns attained by each student throughout the simulation, ensuring a nuanced and dynamic representation of the learning process. This meticulous approach enhances the fidelity of the model and contributes to a more comprehensive understanding of student learning within the specified course context.

The input information for the Teacher Agent comprises course-specific teaching

content, delineated in terms of knowledge items, and a corresponding schedule sequence. This sequence systematically encapsulates the teaching status of the instructor for each day within a predefined timeframe. More precisely, it indicates whether a new lesson is scheduled for a given day in the course. In the event of a scheduled lesson, the sequence specifies the particular content to be presented on that day. The Teacher Agent utilizes this information to effectively manage and deliver course content, contributing to a structured and organized educational environment within the simulation framework.

The input information of the Student Agent includes both individual students' cognitive abilities and the distribution coefficients representing the collective cognitive abilities of all students. At the simulation's outset, each Student Agent's cognitive ability is initialized as a float number within the range of 0 to 1. While various factors may influence individual students' cognitive abilities, research suggests that, under certain conditions, a normal distribution is plausible (Peng & Kievit, 2020). Similarly, human IQ test results often exhibit normal distribution characteristics, with the majority concentrated in the middle levels, while extreme levels gradually impact the speed, depth, and quality of learning. Therefore, for this study, it is assumed that the overall cognitive abilities of all students conform to a normal distribution. Each Student Agent is assigned a random cognitive ability using a Gaussian function, with the mean (μ) and standard deviation (σ) as input parameters for the model.

The LTM Agent is conceptualized as a directed weighted acyclic graph, wherein nodes represent the knowledge concepts or items associated with the course content, and edges symbolize the quantitative relationships between these concepts. Both vertices and edges are systematically organized within a list structure. At the outset, each Student Agent enters the learning environment without possessing any of the knowledge to be acquired during the course. Consequently, a singular virtual node is introduced, serving as the common initial vertex of the LTM. This virtual node represents the Student Agent's prior knowledge, and as the learning progresses, additional nodes are dynamically incorporated into the graph to signify the acquisition

and interconnections of new knowledge concepts. The utilization of a directed acyclic graph framework facilitates a comprehensive representation of the evolving structure of the student's long-term memory, capturing the relationships and dependencies between different knowledge items in the context of the course curriculum.

3.3 Data Processing and Collection

The data gathered for this study comprises a comprehensive list of knowledge attributes that each student agent has mastered by the conclusion of every simulation round. This inclusive approach not only enables the tracking of individual students' learning progress but also unveils the evolving cognitive conditions within the simulated educational environment.

The matrix's columns align with the number of knowledge items covered in the course, while its rows correspond to the number of simulation cycles performed. Each row records the student's learning progress of the studied knowledge concepts after one time step in the simulation. Within the matrix, each element is a vector with two components, signifying the student's engagement with a specific knowledge item. The first component denotes whether the knowledge item has been taught (*0* indicates not taught, *1* indicates taught), while the second component indicates whether the student has mastered the knowledge item (*0* signifies not mastered, *1* signifies mastered).

Illustrated by a sample 9×6 matrix, documenting the learning of 9 knowledge items over 6 cycles by a particular Student Agent, the initial state in the first row reflects that all knowledge items are not taught and not mastered (0, 0). Subsequent rows depict the evolving mastery states after each simulation cycle. For instance, considering the element at row index 3 and column index 2, the vector (1, 1) indicates that by the end of the 3rd simulation cycle, knowledge *A2* was taught through a new lecture session, and the students have successfully mastered it. Conversely, an element like *[1, 0]* at row index 4 and column index 3 implies that at the end of the 4th simulation cycle, knowledge *A3* was taught, but the students have yet to master it. It is crucial to note that the events executed by the Student Agent in each simulation cycle exert an influence on the status of each knowledge point, introducing dynamic interactions within the learning process.

	A0	A1	A2	A3	A4	A5	A6	A7	A8
cycle-0	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]
cycle-1	[1, 1]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]
cycle-2	[1, 1]	[1, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]
cycle-3	[1, 1]	[1, 1]	[1, 1]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]
cycle-4	[1, 1]	[1, 1]	[1, 1]	[1, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]
cycle-5	[1, 1]	[1, 1]	[1, 1]	[1, 1]	[0, 0]	[1, 0]	[0, 0]	[0, 0]	[0, 0]

Furthermore, the scope of data collection extends to the foundational state of items within the LTM, offering a snapshot of existing knowledge elements. Additionally, the state of item associations is documented, providing a nuanced understanding of the intricate interconnections within the cognitive framework. Such a thorough approach to data collection proves instrumental in unraveling the intricacies inherent in the teaching and learning processes within the secondary school curriculum. It sets the stage for a comprehensive analysis of the simulation results, allowing for a nuanced exploration of the multifaceted educational landscape.

Moving forward, a detailed explanation of the vertices and edges within the LTM is provided. Within the vertex list, each vertex is structured as a vector with four components: *[v1, v2, v3, v4]*. The initial component, *v1*, serves as the motivation score assigned to the corresponding knowledge item. A higher motivation score signifies that the student harbors greater enthusiasm and motivation to comprehend the associated concept, while a lower score suggests a comparatively reduced motivation level. The motivation score is computed using the formula presented in Equation 1, elucidating the quantitative relationship between the student's motivation and the specific knowledge concept under consideration. This approach contributes to a nuanced understanding of the student's engagement and motivation dynamics within the learning process, enriching the realism and sophistication of the LTM framework.

$$m_X^{A(t)} = \sum_{Y \in SCx} \left(\frac{1}{l_{XY}^{A(t)}}\right) \qquad (1)$$

Where item *X* represents a vertex denoting a knowledge concept in LTM Agent. The motivational score for item *X* at time step *t*, $m_x^{A(t)}$ reflects Agent A's motivation for understanding that specific concept. *SCx* is the set of concepts connected to item *X*,

encompassing both in- and out-points of the vertex *X* within the LTM. The edge *XY* connects *item X* and *Y*, establishing a relationship between these knowledge concepts. The length of the confusion interval of the XY edges at time t, $l_{xy}^{A(t)}$ denotes the information about the weights on the relevant concepts and their edges, the meaning of which will be further explained and illustrated later.

The second component serves as an indicator of whether the item node has been reviewed in the current simulation cycle: a value of 0 denotes it hasn't been reviewed, while a value of 1 signifies that it has. The default value is set to 0. The third component represents the number of days elapsed since the node was last utilized. The fourth component is the identifier of the knowledge item, aligning with the corresponding identifier in the Course Agent. As an illustration, an element in a vertex list may be expressed as *[5.2, 0, 3, 6]*, signifying that the item with identifier 6 possesses a motivation score of *5.2*, has not been reviewed in the current simulation round, and it has been 3 days since it was last selected for repeat learning.

Within the edge list, each element signifies an edge, structured as a vector with three components. The first component designates the start item identification of the edge, the second component denotes the end item identification, and the third component represents the weight of the edge, namely the confusion interval length. This length is a numerical value ranging from 0 to 1, reflecting the degree of mastery of the relevant knowledge. A higher value implies weaker mastery, making the knowledge more susceptible to forgetting, while a smaller value indicates solid mastery. For instance, considering an edge denoted as e with vector information [0, 1, 0.21], it connects the starting point identified as 0 to the end point identified as 1, and possesses a confusion interval length of 0.21.

Following the ULM definition of the repetition learning process, the repetition of knowledge by the student leads to a reduction in the confusion length value associated with conceptual connections in the LTM. Conversely, items and their connected relationships that have not undergone repetition for an extended duration result in an increase in the confusion interval length value. Throughout each simulation cycle in the MAB-SCD framework, the state of each edge in the LTM Agent dynamically

evolves in response to the passage of time and student-related behaviors. The update operations in three distinct scenarios are elaborated below.

Scenario-1 involves the update operation when a Student Agent reviews item *X* within a simulation cycle. At this juncture, the confusion interval length of *x* is modified, assuming that the edge with item *X* as either the start or end point is represented by x. The formula governing the update of the confusion interval length for an edge can be deduced from Equation (2)(Chiriacescu et al.,2003).

$$w_{XY}^{L(t)} = \frac{cic \cdot \left[f(X) \cdot m_X^{L(t)} + f(Y) \cdot m_Y^{L(t)} \right] \cdot w_{XY}^{T(t)} + w_{XY}^{L(t-1)}}{cic \cdot \left[f(X) \cdot m_X^{L(t)} + f(Y) \cdot m_Y^{L(t)} \right] + 1} \times 2 \qquad (2)$$

In the Equation 2, *XY* denotes the edge connecting vertex *X* to *Y*, with $W_{XY}^{L(t)}$ and $W_{XY}^{L(t-1)}$ representing the lengths of the confusion intervals at simulation time steps *t* and *(t-1)* respectively. The motivation scores of the student agent for items *X* and *Y* at simulation time t are denoted as $m_X^{L(t)}$ and $m_Y^{L(t)}$ respectively. $W_{XY}^{T(t)}$ is the instantiation weighting coefficient of the edge *XY* passed to the LTM, expressed as a constant value. The learning coefficient, *cic*, is a constant value presented in the equation. Additionally, the function *f* returns a Boolean result based on whether a given knowledge concept item is accessed for review in the current simulation round. It returns *1* if the item was accessed for review and *0* otherwise.

Scenario-2 involves the effect of the association between items characterizing knowledge concepts. When a student repeats the learning of *item-X*, it not only enhances the mastery of *X* but also influences the mastery of other items related to *X*, specifically its neighbors. Considering the logical structure in LTM, both the weight values of the *edge-c* where *X* is located and the *edge-x* adjacent to *edge-c* should undergo changes. We term this effect as the confusion interval length extension effect resulting from the review of knowledge points. To model this situation, we leverage the spreading activation instantiation architecture proposed by Chiriacescu et al. (2003) and calculate the updated values of the confusion interval lengths of edge x up to edge c using Eqs. (3), (4), and (5).

$$l_x^{A(t)} = l_x^{A(t-1)} - sf \cdot mf \cdot cil \qquad (3)$$

$$sf = 1 - \frac{d(c,x)}{D} \tag{4}$$

$$mf = f(X) \cdot (m_x - AT) + f(Y) \cdot (m_Y - AT) \tag{5}$$

In Equation 3, $l_x^{A(t)}$ and $l_x^{A(t-1)}$ denote the length of the confusion interval for edge x at time steps t and $(t-1)$, respectively; sf stands for spread factor, which can be computed by Equation 4; mf is the motivation factor, can be computed and obtained by Equation 5; and cil is the learning coefficient that affects the change of the length of the confusion interval in the simulation time step, and the value of which can be set as a constant.

In Equation 4, $d(c,x)$ is the graph distance from *edge-c* to *edge-x*; D is a normalisation factor, which is considered to be the maximum value set for the distance between a pair of edge extension connections when taking into account the spreading activation effect of edges in the LTM.

In Equation 5, f is a function with the same function as in Eq. 2; m_X and m_Y are the motivation scores of item X and Y, respectively (computed from Eq. 1); and AT represents the learner's awareness threshold constant.

Scenario-3 involves the situation in the LTM where, if a consecutive time steps passes without the review of a knowledge concept item, surpassing the model-defined time threshold *forget_day,* the weight values of the edges associated with that item undergo an update. The *Equation 6*(Chiriacescu et al.,2003) is utilized to calculate the length of the confusion interval for the edge where *item-X* is situated.

$$\underset{XY}{A}(t) = l_{XY}^{A(t-1)} \cdot e^{r_{dec}} \tag{6}$$

where X represents the item subjected to the knowledge decay mechanism, and Y is the item connected to X. The lengths of the confusion intervals between edges XY at time steps t and $(t-1)$ are denoted as $l_{XY}^{A(t)}$ *and* $l_{XY}^{A(t-1)}$ respectively. The parameter e is a natural number. Additionally, the decay rate of knowledge, denoted as r_{dec}, signifies the rate of growth of confusion intervals and is set as an experimental parameter, maintaining a constant value between *0* and *1*.

Moreover, if the confusion interval length value of an edge reaches or exceeds *1*, the LTM Agent initiates a deletion operation. This entails removing the subsequent

knowledge points linked to this edge from both the vertex list and the edge list in the LTM, signifying that the student is executing a forgetting operation on the associated knowledge concept.

3.4 Output

At the conclusion of the designated simulation time-steps, the outputs the simulation system depict the academic status of students, both individually and collectively, as influenced by the course instruction. Throughout each simulation time step, the records encompass the amount of mastered knowledge, the achieved progression level, and the mastery state matrix for all knowledge items related to each student Agent. These details comprehensively capture the learning journey of individual students. The comprehensive distribution of students' academic proficiency levels can be represented by tracking the knowledge acquisition of all students and noting the percentage of students at each learning level.

4. Pseudo-code for major functions

4.1 Pseudo-code for Generating the ideal mastery pattern

The following pseudo-code describes a procedure called get_master_pattern that implements the operation of B1 from two matrices, knowledge_reachable_matrix and knowledge_mastered_matrix. Loops and Boolean operations are used in the procedure, which includes column fetching, matrix creation and Boolean addition operations.

```
procedure get_master_pattern():
    m = 0
    for i_col = 0 to len_key - 1 do
        for j_col = (i_col + 1) to (len_key + m - 1) do
            a = column_at(knowledge_reachable_matrix, i_col)
            b = column_at(knowledge_mastered_matrix, j_col)
            c = create_matrix(1, len_key)
            for k = 0 to len_key - 1 do
                k_col = (a[k] or b[k])
                if k_col then
                    c[0, k] = 1
```

```
            else
                c[0, k] = 0
```

4.2 Pseudo-code for Teacher's procedure of teaching a new lesson

The teacher implemented a new lecture consisting of two behaviours, B2 and B3, and the corresponding pseudo-code is shown below.

```
plan choose_item_to_student intention:
    determine the items to be taught based on my_teach_schedule;
    loop item_i over: my_teach_schedule
        item_location <- item_location + 1;
        if (item_i != "#")
            //"#": ending marker of the lecture content "#".
            add item_i to: item_teach;
         else
            break;

plan teach_knowledge_to_student intention:
    if (empty(current_knowledge_teaching))
        establish a sub-intention to determine the content;
        Suspend the execution of the current intention;
    else
        if (current_knowledge_teaching[0] = '*')
            class_state <- false;
        if (the current lecture content is not empty)
            class_state <- true;
            count the "real" items of the current lecture content;
            Statistic the number of knowledge items already taught;
```

4.3 Pseudo-code for Students' procedure of learning

The process of learning new knowledge through a new lecture is B4, and the corresponding pseudocode is as follows.

```
plan learn_new_plan intention:
    get the content for this learning session
    new_item_learn <- math_teacher.count_item_taught;
    if (new_item_learn != null)
        modify the relevant item's first entry;
        do modify_item_learned1(new_item_learn);
        modify the relevant item's second entry;
        reset new_item_learn to nil
```

The process by which students acquire new knowledge that they have not mastered through independent learning is B5, and the corresponding pseudocode is as follows.

```
plan review_plan_unmastered intention:
    if (item_review != null)
        pause the current intention;
        add a new sub-intention;
    else
        randomly determine the number of knowledge points to review;
        modify the status of reviewed attributes;


plan choose_current_item_review1 intention:
    determine the row and column indices of the knowledge point to review;
    store the knowledge point in target_item_review;
    list unmastered_knowledge;
    store all learned but unmastered knowledge points in unmastered_knowledge
```

The process by which students consolidate acquired knowledge through independent revision is B6, corresponding to the following pseudo-code.

```
plan choose_current_item_review2 intention:
    retrieve the list of mastered knowledge from the LTM by motivation score;
    list mastered_knowledge <- my_LTM.vertices_orderby_score;
    if (length(mastered_knowledge) = 0) {
```

```
        do remove_intention(review_mastered_knowledge, true);
    } else {
        randomly choose n items from the sorted list;
        Call:plan learn_new_plan intention;
```

4.4 Pseudo-code for management operations in LTM Agent

The behaviour of LTM in relation to the management of conceptual attributes and their relationships corresponds to B7, which contains a series of procedures such as adding, deleting and modifying. The corresponding pseudo-codes are as follows.

```
    procedure insert_LTM(int item_mastered) :
        insert item_mastered into the list of vertices in the LTM：
          do add_vertex_to_LTM(item_mastered);
        modify the state of the vertex
          do modify_vertex_state(item_mastered);
        find and store the incoming edges for the item_mastered vertex
          list incoming_edges <- find_incoming_edges(item_mastered);
          do store_incoming_edges(item_mastered, incoming_edges);
        modify the state of edges related to item_mastered
          do modify_related_edges_state(item_mastered);


procedure LTM_modify(int v_r):
    sub-procedure action_weaken(int x) {
        //find edges with x as the source
          list outgoing_edges <- find_outgoing_edges(x);
            loop i over: outgoing_edges {//process each outgoing edge
              use formula 7 to modify the confusion interval length of these edges;
        }

       //find edges with x as the target
         list incoming_edges <- find_incoming_edges(x);
            loop j over: incoming_edges {// process each incoming edge
```

```
                use formula 7 to modify the confusion interval length of these edges;
            }
        }

        sub-procedure LTM_revise(int v_r) {
            // modify the state of vertex v_r
            LTM_vertices[v_r][1] <- 1;
            LTM_vertices[v_r][2] <- 0;
            // find edges associated with vertex v_r
            list edges <- find_associated_edges(v_r);
            // process each edge
            loop k over: edges {
                use formulas 4, 5, 6 to modify the edges;
             }
        }

procedure delete_from_LTM:
    // find all vertices that meet the deletion criteria
        list d_items <- find_vertices_to_delete();
        loop i over: d_items { // process each vertex for deletion
            // perform a depth-first traversal to delete the vertex and associated edges
              depth_first_delete(i);
        }

    sub-procedure depth_first_delete(int vertex):
        // Base case: if the vertex is null or already marked for deletion, return
        if (vertex == null or marked_for_deletion(vertex)) {
            return;
        }
        // Mark the vertex for deletion
```

```
mark_vertex_for_deletion(vertex);
// Find outgoing edges from the current vertex
list outgoing_edges <- find_outgoing_edges(vertex);
// Process each outgoing edge
loop j over: outgoing_edges {
    // Recursively delete the target vertex of the outgoing edge
    depth_first_delete(get_target_vertex(j));
    // Delete the outgoing edge
    delete_edge(j);
}
// Delete the current vertex
delete_vertex(vertex);
```

5. Platform and Environment Requirements

The simulation model developed on the GAMA(Drogooul et al., 2013) platform (*Version 1.9)* adheres to specific technical specifications to ensure optimal functionality and compatibility. The model is coded in GAML (GAMA Modeling Language), the native language of the GAMA platform , leveraging its agent-based modeling capabilities. GAMA provides a user-friendly and extensible environment for developing, running, and analyzing simulations.

The platform and environment requirements for running the simulation model include a system with sufficient computational resources, such as a standard personal computer with a multi-core processor and ample RAM. The GAMA platform is compatible with major operating systems, including Windows, macOS, and Linux, providing flexibility in deployment.

To facilitate collaboration and version control, the model is organized into modular components, enhancing code readability and maintainability. The GAMA platform's inherent support for agent-based modeling simplifies the representation of entities and interactions within the educational environment, making it well-suited for our simulation objectives.

In conclusion, the GAMA-based simulation model is designed with specific

technical specifications to run on standard computing hardware, and its compatibility with multiple operating systems ensures widespread accessibility. The modular organization of the model and integration of external libraries contribute to its robustness and fidelity in simulating student cognitive development within educational environments.

6. Model Validation

Model validation is a crucial step in ensuring the reliability and accuracy of the simulation model developed on the GAMA platform. To validate our model, we employed a multi-faceted approach encompassing both face and predictive validity. Face validity was achieved by consulting domain experts and educators to ascertain whether the model's structure, mechanisms, and parameters align with real-world educational processes. Their feedback and insights were instrumental in refining the model to better reflect the complexities of student cognitive development within educational environments.

Furthermore, sensitivity analyses were conducted to assess the robustness of the model. We systematically varied input parameters and assessed the impact on the model's output. The consistency of the model's behavior across a range of parameter values contributed to its validation.

In summary, our validation process involved collaborative expert input, empirical scenario testing, and sensitivity analyses, ensuring that the GAMA-based simulation model effectively captures the dynamics of student cognitive development in educational settings.

7. Conclusion and Future Enhancements

The MAB-SCD system operates as a tool offering a exploration of the intricate interplay between cognitive development and educational instruction. The integration of the BDI cognitive architecture and Cognitive Diagnostic Assessment theories underscores its robust foundation, making it an aid to assist educational researchers in gaining insight into students' learning experiences.

In the subsequent study, a series of scenario-based tests will be conducted to enable calibration and validation of the model by comparing its outputs with the

expected outcomes derived from existing educational literature and empirical data. Specifically, this will include assessing whether the model's ability to simulate student knowledge acquisition, retention and progression is consistent with established educational theory and observed student behaviour. Based on the results of these tests, the model was iteratively adapted to ensure that it could generate meaningful and reasonable predictions.

8. References

Caporale, N., & Dan, Y. (2008). Spike timing–dependent plasticity: a Hebbian learning rule. *Annu. Rev. Neurosci.*, *31*, 25-46.

Chiriacescu, V., Soh, L. K., & Shell, D. F. (2013, July). Understanding human learning using a multi-agent simulation of the Unified Learning Model. In *2013 IEEE 12th International Conference on Cognitive Informatics and Cognitive Computing* (pp. 143-152). IEEE.

Drogoul, A., Amouroux, E., Caillou, P., Gaudou, B., Grignard, A., Marilleau, N., Taillandier, P., Vavasseur, M., Vo, D.A., Zucker, J.D.: Gama: multi-level and complex environment for agent-based models and simulations. In: AAMAS. pp. 1361–1362 (2013)

Howden, N., Rönnquist, R., Hodgson, A., & Lucas, A. (2001, May). JACK intelligent agents-summary of an agent infrastructure. In *5th International conference on autonomous agents* (Vol. 6).

Kennedy, W. G. (2011). Modelling human behaviour in agent-based models. In *Agent-based models of geographical systems* (pp. 167-179). Dordrecht: Springer Netherlands.

Leighton, J. P., Gierl, M. J., & Hunka, S. M. (2004). The attribute hierarchy method for cognitive assessment: A variation on Tatsuoka's rule-space approach. *Journal of educational measurement*, *41*(3), 205-237.

Ormazábal, I., Borotto, F. A., & Astudillo, H. F. (2021). An agent-based model for teaching–learning processes. *Physica A: Statistical Mechanics and its Applications*, *565*, 125563.

Peng, P., & Kievit, R. A. (2020). The development of academic achievement and

cognitive abilities: A bidirectional perspective. *Child Development Perspectives*, *14*(1), 15-20.

Pokahr, A., Braubach, L., & Lamersdorf, W. (2005). Jadex: A BDI reasoning engine. *Multi-agent programming: Languages, platforms and applications*, 149-174.

Rohde, T. E., & Thompson, L. A. (2007). Predicting academic achievement with cognitive ability. *Intelligence*, *35*(1), 83-92.

Tatsuoka, K. K. (2013). Toward an integration of item-response theory and cognitive error diagnosis. In *Diagnostic monitoring of skill and knowledge acquisition* (pp. 453-488). Routledge.

Tikhomirova, T., Malykh, A., & Malykh, S. (2020). Predicting academic achievement with cognitive abilities: Cross-sectional study across school education. *Behavioral sciences*, *10*(10), 158.