Extending Adaptive Spacing Heuristics to Multi-Skill Items

Benoît Choffin
Université Paris-Saclay*
benoit.choffin@lisn.upsaclay.fr

Fabrice Popineau
Université Paris-Saclay*
fabrice.popineau@lisn.upsaclay.fr

Yolaine Bourda
Université Paris-Saclay*
yolaine.bourda@lisn.upsaclay.fr

Adaptive spacing algorithms are powerful tools for helping learners manage their study time efficiently. By personalizing the temporal distribution of retrieval practice of a given piece of knowledge, they improve learners' long-term memory retention compared to fixed review schedules. However, such algorithms are generally designed for the pure memorization of single items, such as vocabulary words. Yet, the spacing effect has been shown to extend to more complex knowledge, such as the practice of mathematical skills. In this article, we extend three adaptive spacing heuristics from the literature for selecting the best skill to review at any timestamp given a student's past study history. In real-world educational settings, items generally involve multiple skills at the same time. Thus, we also propose a multi-skill version for two of these heuristics: instead of selecting one single skill, they select with a greedy procedure the most promising subset of skills to review. To compare these five heuristics, we develop a synthetic experimental framework that simulates student learning and forgetting trajectories with a student model. We run multiple synthetic experiments on large cohorts of 500 simulated students and publicly release the code for these experiments. Our results highlight the strengths and weaknesses of each heuristic in terms of performance, robustness, and complexity. Finally, we find evidence that selecting the best subset of skills yields better retention compared to selecting the single best skill to review.

Keywords: adaptive spacing, skill selection heuristics, knowledge components, multi-skill learning items

1. Introduction

Considering that learning new knowledge often builds on older knowledge and requires significant cognitive efforts, improving long-term memorization is critical in education. Fortunately, cognitive scientists have uncovered several learning strategies that help learners organize their study time and improve their memory retention at a small cost. In particular, *spaced repetition* consists of temporally distributing learning episodes instead of learning in a single massed study session (Roediger III and Karpicke, 2011; Cepeda et al., 2008). The memorization benefits of spaced repetition over massed study are called the *spacing effect*, and spaced repetition has been

*Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire Interdisciplinaire des Sciences du Numérique, 91190, Gif-sur-Yvette, France

advocated as one of the best tools to improve public instruction by cognitive scientists (Weinstein et al., 2018; Dunlosky et al., 2013).

However, even if the benefits of spaced repetition compared to massed learning are clearly established, how in practice should students schedule their reviewing sessions? Considering that the time delay between the initial acquisition and the subsequent reviews largely determines the magnitude of the spacing effect (Cepeda et al., 2008), this issue is of the highest importance in education.

Recent scientific advances have seen the development of adaptive spacing schedulers (Lindsey et al., 2014; Tabibian et al., 2019). By focusing on the items[1] that would benefit the most from being reviewed, these tools are able to significantly improve memory retention in the medium and long terms, compared to fixed spacing schedules (Mettler et al., 2016; Lindsey et al., 2014).

Nonetheless, these algorithms currently only work for the pure learning and memorization of simple pieces of knowledge, such as factual knowledge (e.g., vocabulary words). To the best of our knowledge, no research work has ever sought to extend these algorithms for learning a set of skills.

Yet, the spacing effect is not limited to vocabulary learning or to the pure memorization of items: the spaced repetition strategy has for instance been successfully applied to the acquisition and generalization of abstract scientific concepts (Vlach and Sandhofer, 2012) and to the practice of mathematical skills in a real educational context (Barzagar Nazari and Ebersbach, 2019). Thus, we investigate in this article the research problem of optimizing long-term mastery of skills with adaptive spacing algorithms.

In this article, we assimilate skills and knowledge components. A knowledge component (KC) is "a description of a mental structure or process that a learner uses, alone or in combination with other knowledge components, to accomplish steps in a task or a problem. [...] A knowledge component is a generalization of everyday terms like concept, principle, fact, or skill [...]."[2]. Examples of KCs include the concept of nested loops in programming (Effenberger et al., 2020) and solving quadratic equations in mathematics (Wang et al., 2020). When we say that an item involves a KC, this means that solving the item requires applying this KC correctly.

Our first contribution is the development of a synthetic experimental framework that relies on simulated student learning and forgetting trajectories to compare the performance of adaptive spacing algorithms. This framework requires using a student model for generating synthetic student behaviors. In this framework, each simulated student reviews a set of KCs by sequentially interacting with an adaptive and personalized spacing system. At each time step, the system adaptively selects an item for the simulated student, e.g. in a math setting "*What is* $\lim_{x\to 0}(\sin x)/x$*?*". The timing of each review session is *not* determined by the algorithm: rather than trying to optimize the timestamp for reviewing a given item, we focus on selecting the optimal KC (or set of KCs) to review *at a given timestamp*. We believe that such settings are more realistic for inter-session scheduling since, in the real world, a student may not always be available for practicing KCs when the algorithm decides it is time. Then, the simulated student answers the item, and the system uses the correctness of the answer to update its belief concerning the student's current and future mastery of the KCs involved by the item. One major difference with standard adaptive spacing algorithms is that an item can involve multiple KCs at the same time. The goal of the algorithm is to optimize mastery of the set of KCs during a future and

[1] An item is a pedagogical activity involving knowledge retrieval from the learner's memory. The notion of item generalizes the notions of exercise, question, test, etc.

[2] https://www.learnlab.org/research/wiki/Knowledge_component

extensive time period by carefully selecting the sequence of KCs for each student. Depending on the strengths and weaknesses of each student, the algorithm may dynamically choose a different sequence of KCs to better suit the student's needs. After selecting a KC (or a set of KCs), the algorithm selects for the student an item that involves the chosen KC or KCs.

Our second contribution is simple and efficient adaptive spacing algorithms for optimizing the retention of a set of KCs. We extend three adaptive spacing heuristics from the literature to our research problem. Most importantly, we propose two multi-KC versions of these heuristics: instead of selecting *one* single KC, they select with a greedy procedure the most promising *subset* of KCs to review.

Our third contribution is the synthetic experiments that we conduct to compare these heuristics using the simulated experimental framework that we proposed. These experiments allow us to showcase the properties of our heuristics in controlled and idealized conditions. Our experiments yield contrasting results for the five heuristics: each has its strengths and weaknesses in terms of student memory retention, robustness to atypical learning and forgetting behaviors, and complexity. Our second main finding is that selecting multiple KCs at once improves over selecting the single best KC.

The rest of the article is organized as follows: in Section 2, we review previous works on adaptive spacing algorithms. Then, in Section 3, we detail DAS3H, the student learning and forgetting model that we used to conduct our simulated experiments. In Section 4, we describe the synthetic experimental protocol that we developed. In Section 5, we present the different KC selection strategies that we implemented. Moreover, we propose a greedy selection procedure for selecting multiple promising KCs at the same time, instead of one single KC. In Section 6, we present our experimental results and discuss them in Section 7. Finally, we conclude the article in Section 8.

2. Related Works

Adaptive spacing algorithms seek to provide each learner with the sequence of item reviews that will benefit their long-term memorization the most, by taking into account their specific needs. These needs are most often inferred by the algorithm from the learner's current practice history (previous recall attempts, outcomes and timestamps of these attempts, response times, etc.). We call these systems *adaptive* because they take into account the ongoing learner's performance to dynamically offer them a review schedule suited to their needs.

In most algorithms, each of these reviews consists of answering a question asked by the adaptive spacing system. Adaptive spacing algorithms therefore use two efficient learning strategies studied in the cognitive science literature: *spaced repetition* because they aim at optimally spacing out the successive reviews of a piece of knowledge, and *retrieval practice* because each review requires the learner to make the effort to remember an item rather than reading the answer directly.

Historically, these algorithms have focused on scheduling reviews for simple pieces of knowledge: vocabulary words in a foreign language (Pimsleur, 1967; Pavlik and Anderson, 2008; Metzler-Baddeley and Baddeley, 2009; Lindsey et al., 2014), locating African countries on a map (Mettler et al., 2016), or learning historical facts. Flashcards are often used to represent items in adaptive spacing algorithms. At first physical (Leitner, 1972), electronic flashcards developed with the advent of computer-assisted instruction in the second part of the 20th century. Electronic flashcards allow more fine-grained adaptive spacing algorithms that will optimize the

teaching sequence for increased long-term memory retention. In this section, we review previous research works on adaptive spacing algorithms. Following Doroudi et al. (2019), we split our literature review into two groups: *model-based* and *model-free* algorithms. Model-based algorithms use the predictions (of correctness, latency, etc.) of a student model that has been fit on performance data to schedule future reviews, whereas model-free algorithms do not rely on such models.

2.1. Model-Based Algorithms

Among model-based algorithms, a popular option is to choose at time t the item for which a review priority score is closest to a given value θ. For instance, Pavlik and Anderson (2008) used an extended version of the ACT-R declarative memory model to build an adaptive scheduler for optimizing item practice (in their case, 180 Japanese-English word pairs) given a limited amount of time within three reviewing sessions. In its original form, ACT-R is capable of predicting item correctness and speed of recall by taking recency and frequency of practice into account. Pavlik and Anderson extended ACT-R to capture the spacing effect as well as item, learner, and item-learner interaction variability. Their adaptive scheduler uses the model estimation of memory strength gain at retention test per unit of time to decide when it is best to present each pair of words to the learner. This algorithm was also used in two classroom experiments (Pavlik et al., 2008).

Van Rijn et al. (2009) compared four algorithms, including the algorithm of Pavlik and Anderson (2008) and two versions of it that use individual performance and latency to further improve spacing personalization. Their goal was to schedule reviews of 20 French-Dutch word pairs, also within a single learning session. Their most sophisticated algorithm (latency-based) outperformed two other algorithms.

Our current work is related to the work of Khajah et al. (2014). In this article, Khajah et al. used computational simulations based on two cognitive models of human memory (Pavlik and Anderson, 2008; Pashler et al., 2009) to compare the efficiency on student long-term memory of two item selection heuristics: μ-*back* and θ-*threshold*. They simulated a learning environment with K chapters (or knowledge components), sequentially introduced a week apart from each other; at each week, students reviewed a previous block by trying to recall it correctly. The μ-*back* strategy selects the item that was seen μ weeks ago and the θ-*threshold* strategy, the item for which the recall probability is closest to a fixed value θ. They argued that this heuristic is consistent with Bjork's notion of "desirable difficulties" in human learning, because it makes the student review an item that is on the verge of being forgotten. They found that a *2-back* strategy (i.e., making students review the item that was introduced two weeks ago) is sufficient for teachers' use without additional educational technology tools. However, they also found that a well-parameterized θ-*threshold* (with θ around 0.4) strategy achieves close-to-optimal student performance. This finding suggests that with access to student data, an algorithm could achieve better student memory retention than fixed schedules.

This θ-*threshold* item selection strategy was successfully employed in a real-world classroom experiment by Lindsey et al. (2014). In this experiment, Lindsey et al. used a student predictive model – the DASH model – to predict student correctness probability on any of the previously introduced items. They chose the item whose estimated probability of correct recall is closest to $\theta = .33$ at a given timestamp.

Eglington and Pavlik (2020) have argued that most adaptive spacing algorithms that use

recall probability to select which item should be reviewed at a given timestamp yield suboptimal schedules because they do not take time costs into account. Items that are more difficult to recall generally take more time to the student, especially because they need to read the correct answer afterwards. Thus, rather than purely considering the learning gain at test time, Eglington and Pavlik proposed to weight the expected learning gain of selecting an item by the time cost of each possible outcome (success or failure). Their algorithm selects the item for which the student model estimate of recall probability is closest to (but still less than) a given threshold. They ran both simulated and real-world experiments, in which participants had to study 30 Japanese-English word pairs for 22 minutes and to complete a test 3 days later. They compared different recall probability thresholds in terms of final test performance and found a much higher optimal recall threshold (0.86) than Khajah et al. (2014).

Some rarer model-based works have chosen instead to rely on myopic optimization, which means that they only consider the expected consequences of their next item selection decision on the performance metric that they are trying to optimize. This is the case for Hunziker et al. (2019) who proposed a greedy item selection algorithm based on discrete optimization for flashcard review scheduling. Given a limited time budget and based on a student's past study history, this algorithm iteratively selects the concept (or item) that will provide the highest expected correctness probability gain over all concepts and all future timestamps. Such a decision requires the use of a student memory model to infer the impact of choosing any concept at time t. In addition to providing performance guarantees for their algorithm, Hunziker et al. test it by interacting with simulated learners and human subjects recruited on Mechanical Turk.

Tabibian et al. (2019) tackled the adaptive spaced repetition problem in a different manner. They formalized it within the Marked Temporal Point Processes framework as a stochastic optimal control problem. The recall probability dynamics and forgetting rates for each item are modelled by Stochastic Differential Equations (SDEs) with jumps. Tabibian et al. analytically solve this optimal control problem for three cognitive models of human memory, including the exponential forgetting curve model (Ebbinghaus, 1885). Contrary to Hunziker et al., their MEMORIZE algorithm considers the whole learning period to choose the optimal timestamps at which each item should be reviewed.

2.2. MODEL-FREE ALGORITHMS

As Reinforcement Learning methods have been used since the beginning of Intelligent Tutoring Systems (Doroudi et al., 2019), they have also been applied recently to optimally schedule reviews in adaptive spacing algorithms. Reddy et al. (2017) used a deep reinforcement learning architecture to tackle the issue of optimally spacing out reviews. They formulated this problem as a Partially Observable Markov Decision Process (POMDP). Their model-free algorithm has only access to three features: the last item id, whether the student managed to recall it, and the time delay between this last interaction and the current timestamp. They compared their algorithm to several baselines, including a variant of the SuperMemo algorithm (Wozniak and Gorzelanczyk, 1994) and the threshold-based policy from Lindsey et al. (2014). They used simulated students, based on three models of human memory, for this purpose. Similar to Tabibian et al., Upadhyay et al. (2018) used deep reinforcement learning with marked temporal point processes to optimally schedule flashcard reviews. One of the advantages of their algorithm is that it is independent of the learner's specific memory model; however, it assumes that the learner can practice the items at any time.

We described above different model-based adaptive spacing algorithms that use a priority score to plan successive item reviews. Similar algorithms, but without any predictive student model, also exist in the literature. For instance, Mettler et al. (2016) compared an adaptive spacing scheduler (coined ARTS for Adaptive Response-Time-based Sequencing) to two fixed spacing conditions. ARTS leverages the student's response time, performance, and number of trials to dynamically compute a priority score for adaptively scheduling item practice. Response time is used as an indicator of retrieval difficulty and thus, learning strength. Then, the item that has the highest score at a given timestamp is proposed to the student.

Simpler model-free adaptive spacing algorithms can also be found in the literature. For instance, the long-standing Leitner system (Leitner, 1972) was one of the first adaptive spacing algorithms. In this system, different boxes are assigned different reviewing rates (e.g., every day, every two days, etc.). When a new item enters the system, it is automatically put into the most frequently reviewed box. Every reviewing session, the learner is sequentially presented with every flashcard of the deck they have to review that day. If they fail to recall the item, it is moved from box of index i to the box below[3] (index $i-1$, more frequently reviewed). If they correctly recall it, it is put into the box above (index $i+1$, less frequently reviewed). In the end, if the learner correctly recalls an item that was previously in the least frequently reviewed box, the flashcard is considered mastered and leaves the Leitner system. The Leitner system can be considered a prototype of future adaptive spacing algorithms: the rationale behind it is that depending on your performance, you will be assigned the flashcards that you need to review the most. A more principled version of the Leitner system has been proposed since then (Reddy et al., 2016): in this work, Reddy et al. modeled the Leitner system as a queuing network and find an approximation of the optimal reviewing schedule with the help of a heuristic.

More recently than the Leitner system, Wozniak and Gorzelanczyk (1994) developed the SuperMemo[4] adaptive spacing algorithm. This algorithm, which has known several subsequent versions throughout the years, combines multiple handcrafted rules to compute for each item the next time that it should be reviewed by the learner. It is used in the open-source Anki and Mnemosyne software and was also used in a real-world experiment by Metzler-Baddeley and Baddeley (2009).

Nonetheless, the lack of theoretical – and sometimes empirical – performance guarantees among adaptive spacing algorithms based on handmade rules has since raised some concerns (Tabibian et al., 2019).

2.3. Limits of the Existing Adaptive Spacing Algorithms

To the best of our knowledge, all adaptive spacing algorithms are designed for learning simple pieces of knowledge, items that involve one single KC. Previous work has already taken into consideration the possibility of clustering several items within a single KC (Khajah et al., 2014; Lindsey et al., 2014) or the possibility that reviewing items can help memorize related concepts (Hunziker et al., 2019), but to the best of our knowledge, none of them has ever addressed the issues of (1) scheduling reviews when *single* items can involve *multiple* KCs at the same time and (2) when the objective is to master this set of KCs and not the set of practice items. Yet, this type of item is not rare in real world educational situations: for instance in language learning or

[3] In an alternative version of the algorithm, it is instead put into the most frequently reviewed box (index 0) after failure.

[4] https://www.supermemo.com/

in mathematics[5].

With this article, we seek to develop adaptive spacing algorithms for optimally scheduling reviews for a set of KCs. The difference between our problem and previous work is twofold:

- The objective is to make the learners master a set of KCs *by practicing* a set of items;
- Each item can involve several KCs at the same time.

3. DAS3H Student Simulator Model

Our experimental protocol for comparing spacing strategies requires a student model to generate student answers to items based on their previous performance on the set of KCs to learn. In this Section, we describe DAS3H (Choffin et al., 2019), the student model that we use in our synthetic experiments for simulating student learning and forgetting trajectories. We use DAS3H to compare every adaptive spacing algorithm that we propose.

In what follows, we will index students by $s \in [\![1, S]\!]$, items by $j \in [\![1, J]\!]$, knowledge components (KCs) by $k \in [\![1, K]\!]$, and timestamps by t. $Y_{s,j,t} \in \{0, 1\}$ gives the binary correctness of student s answering item j at time t. σ is the logistic function:

$$\forall x \in \mathbb{R}, \sigma(x) = 1/(1 + \exp(-x)). \tag{1}$$

An item may involve one or more KCs, and this information is synthesized within a binary q-matrix (Tatsuoka, 1983):

$$\forall (j, k) \in [\![1, J]\!] \times [\![1, K]\!],\ q_{jk} = \mathbb{1}_{k \in KC(j)}. \tag{2}$$

$KC(.)$ takes as input an item index j and outputs the set of KC indices involved by item j. Thus, we assume that items are not independent from each other (their relationships are described within a q-matrix). On the other hand, we make the simplifying assumption that KCs are independent from each other and that practicing a KC does not impact another KC's mastery. Our experimental framework and our algorithms could be easily adapted to the situation where KCs can be related to each other, but this goes beyond the scope of this article.

DAS3H stands for item Difficulty, student Ability, Skill and Student Skill practice History. In its simplest formulation (with a feature embedding dimension of $d = 0$), DAS3H formulates the student correctness probability as:

$$\mathbb{P}\left(Y_{s,j,t} = 1\right) = \sigma\left(\alpha_s - \delta_j + \sum_{k \in KC(j)} \beta_k + h_\theta\left(\mathrm{t}_{s,j,t}, \mathrm{y}_{s,j,t}\right)\right) \tag{3}$$

Thus, the correctness probability of student s on item j at time t depends on their ability α_s[6], the item difficulty δ_j and the sum of the easiness biases β_k of the KCs involved by item j. The higher α_s or β_k are, or the lower δ_j is, the higher the correctness probability.

This probability also depends on the temporal distribution and the outcomes of past practice, synthesized by h_θ. This h_θ temporal module is in charge of modeling the learning and forgetting

[5]Items in the *Algebra 2005-2006* dataset (Stamper et al., 2010) have for instance an average of 1.363 KC labels.

[6]α_s can therefore be seen as the learner's initial proficiency, without any practice on the KCs of the curriculum.

of KCs based on the history of the learner's practice. It is given at Equation 4 :

$$h_\theta(\mathrm{t}_{s,j,t}, \mathrm{y}_{s,j,t}) = \sum_{k \in KC(j)} \sum_{w=0}^{W-1} \theta_{k,2w+1} \log(1 + c_{s,k,w}) + \theta_{k,2w+2} \log(1 + a_{s,k,w}). \tag{4}$$

with:

- $\mathrm{t}_{s,j,t}$ the timestamps of student s's past interactions with the KCs involved by item j. This includes in particular the current timestamp t;
- $\mathrm{y}_{s,j,t}$ the binary outcomes of these interactions. Of course, this does not include the binary outcome $Y_{s,j,t}$ since this is what we are trying to infer;
- $w \in [\![0, W-1]\!]$ the time window indices;
- $c_{s,k,w}$ the counter of past correct answers of learner s on items involving KC k in time window w;
- $a_{s,k,w}$ the counter of past attempts of learner s on items involving KC k in time window w;
- $\theta \in \mathbb{R}^{K \times 2W}$ the matrix of coefficients associated with the success and attempt counters. These coefficients describe the learning and forgetting curve of a KC k.

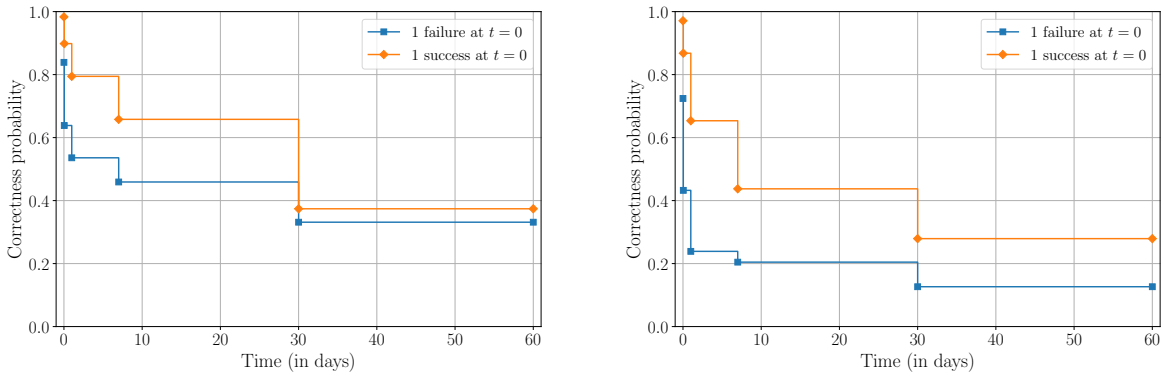
We can see that h_θ is using a set of W expanding time windows to model the impact of the temporal distribution of past practice and the outcomes of these attempts on current and future correctness probability.

DAS3H allows the influence of past practice to vary from one KC to another. Concretely, DAS3H estimates a learning and forgetting curve per KC: these are the $\theta_{k,w}$ coefficients that describe the evolution of the correctness probability according to a learner's practice history (interaction timestamps and outcomes). The time module h_θ then combines these curves by summing them. Intuitively, h_θ can be interpreted as a sum of memory traces, one for each KC k involved by the item j. Please note again that an item can involve more than one KC in our framework. For simplifying the interpretation of the $\theta_{k,w}$ parameters, we slightly modified the formulation of h_θ by replacing the difference of log counts by a sum inside each time window. Please refer to Choffin et al. (2019) for more details about DAS3H.

Figure 1 displays two random forgetting curves generated from DAS3H according to our scheme (cf. Subsection 4.2). In each case, curves represent the temporal evolution of the correctness probability of a hypothetical student of average proficiency answering an average difficulty item involving only a single KC. At $t = 0$, the student makes one attempt on that KC: blue curves represent memory decay if the outcome of this attempt is negative (a failure), orange curves represent memory decay if the outcome is positive (a success).

DAS3H was tested on three large real-world educational datasets and outperformed by a substantial margin four state-of-the-art student models (Choffin et al., 2019). We chose DAS3H as our student simulator model[7] both because (i) of its superior fit to real-world educational data

[7]Notice however that our simulation framework is independent from the specific student model choice. Any student model that models learning and forgetting on KCs and that allows items to depend on multiple KCs can be used.



(a) Slow forgetting

(b) Rapid forgetting

Figure 1: Example forgetting curves generated from our DAS3H simulating model. On each plot, two situations are presented: in blue, an average student practices the KC at $t = 0$ but fails; in orange, they succeed. These curves were generated according to our main experimental protocol, described in Subsection 4.2.

compared to other student models and (ii) it was to the best of our knowledge the only student model from the literature that incorporated both item-KCs relationships and the forgetting effect in its structure to predict the correctness of future student answers. DAS3H parameters also have a direct cognitive interpretation, which makes it easier to simulate realistic learning trajectories and to control the learning/forgetting behaviors of the synthetic students.

In this article, we choose an embedding dimension of $d = 0$ for DAS3H[8]. Finally, we use the set of time windows $\{1/24, 1, 7, 30, +\infty\}$ (in days).

4. Experimental Protocol for Comparing Adaptive Spacing Heuristics

In this Section, we describe the synthetic experimental protocol that we developed to compare adaptive spacing heuristics. Following previous works (Reddy et al., 2017; Khajah et al., 2014; Tabibian et al., 2019), we decided to simulate synthetic learning and forgetting trajectories of learners to whom we assigned one or the other of these spacing strategies. Using simulations allows us to showcase the properties of our heuristics in a controlled and idealized environment on large populations of simulated students.

We detail in Subsection 4.2 how we generate the different parameters of our simulations, and we finally present in Subsection 4.3 the different metrics used to compare the performance of the spacing algorithms.

4.1. Simulating Learning and Forgetting Trajectories

We simulate a simple yet realistic learning and reviewing process, close to the synthetic experimental design from Khajah et al. (2014) and the real-world experimental design from Lindsey

[8]DAS3H was originally formulated within the Knowledge Tracing Machines framework (Vie and Kashima, 2019) to enrich the model by estimating multidimensional feature embeddings and modeling pairwise interactions between these embeddings. We showed (Choffin et al., 2019) that multidimensional feature embeddings did not consistently improve model predictive performance.

et al. (2014). This process leverages both *spacing* and *testing* effects from the cognitive science literature. We wanted it to reflect real-life educational situations where teachers would like to use an adaptive spacing algorithm in their class during a dedicated reviewing session. Nonetheless, our experimental protocol could be easily adapted to other situations, provided that the optimization concerns the KC to review at a given timestamp, not the best timestamp to review KC k (cf. Section 1).

Algorithm 1 Learning and Reviewing Process for a Single Student

procedure LEARN AND REVIEW(KC selection strategy, item selection strategy, K KCs, DAS3H parameters, r items per review session, retention delays)
  for *week* **from** 1 *to* K **do** ▷ Learning/reviewing phase
    Learn initially current week's KC: add 1 attempt to that KC;
    if *week* $>$ 1 **then**
      for i **from** 1 *to* r **do**
        $k^\star \leftarrow$ SELECT SKILL(KC selection strategy)
        $j^\star \leftarrow$ SELECT ITEM($k^\star$, item selection strategy)
        Solve item $j^\star$ and update student's practice history;
    Measure *ACP* at *week*;
  for each τ **in the** *retention delays* **do** ▷ Retention phase
    Measure *ACP* at $K + \tau$;
  return ACP_L and ACP_R
end procedure

Algorithm 1 describes the simulated learning and reviewing process that each learner follows: it generates a learning and reviewing trajectory for a single learner according to a KC selection strategy and some generated simulation parameters. We detail this algorithm in what follows.

The learner starts with no known experience in any of the K KCs of the curriculum that they have to master. Their potential prior knowledge is synthesized in their parameter α_s which, combined with the item difficulty and KC easiness biases δ_j and β_k, gives the probability that a learner s with no previously known practice on the KCs involved by the item j, can answer it correctly (cf. Equation 3).

Every week (i.e., every seven days), a new KC is introduced to the learner. The introduction order of the KCs is fixed in advance: KC i is first introduced and learned in week i. Contrary to the timestamps t, we choose to make the weeks' indices start at 1[9] to ease the interpretation of our results. When a new KC is introduced to student s, we synthesize this additional knowledge by adding a single attempt at week w to s's learning history[10] on KC $k_{current}$.

Then, right after the introduction of the new KC k, the learner is presented with a fixed amount of r review items that they try to solve sequentially. The parameter r is fixed for all simulations, to ensure that every learner in every spacing strategy was assigned the same number of practice items. These items are used to review the previously introduced KCs: the KC(s) are first chosen by the spacing heuristic that s was associated to (SELECT SKILL routine in Algorithm 1) and an item involving the chosen KC(s) is finally determined by the item selection strategy (SELECT ITEM routine in Algorithm 1). Neither the KC that has been introduced the same week

[9]Week 1 therefore ranges from $t = 0$ to $t = 604{,}800$ (in seconds).
[10]This history is therefore empty for KC k before this initial attempt.

Algorithm 2 Experimental Protocol for Comparing Strategies on Simulated Students

procedure SIMULATION(KC selection strategies, item selection strategies, N runs, K KCs, S students, J items, r items per review session, retention delays)
  for i **in** 1 *to* N **do**
    Generate the q-matrix;
    Generate the DAS3H parameters: α_s, β_k, δ_j, $\theta_{k,w}$;
    for each student **in** $\{\alpha_s\}_{s\in[\![1,S]\!]}$ **do** ▷ Each student loops through *all* strategies
      for each KC selection strategy **do**
        for each item selection strategy **do**
          $\sigma \leftarrow$ LEARN AND REVIEW(KC selection strategy, item selection strategy, K, DAS3H parameters, r, retention delays)
          Clear the learning history ($c_{s,k,w}$ and $a_{s,k,w}$) of s;
    Store run i results;
  return Average performance metrics $\bar{\sigma}$ for each (KC selection strategy, item selection strategy) pair;
end procedure

nor the subsequent KCs (not yet introduced) can be reviewed. Therefore, review sessions start at week 2. The learner then solves the item, and their interaction history is updated based on whether or not they correctly answered the item. We assume that the learners' correctness on items is binary and deterministic (i.e., if $\mathbb{P}$("correct answer") > 0.5, then the learner's answer will be correct), and that the time taken to solve an item does not vary from one learner (or item) to another. Contrary to Pavlik and Anderson (2008), learners are assigned a fixed number of review items to solve each week, regardless of the time taken to solve them.

The only difference between the learning paths of two learners lies in the specific items that each learner solves during the review sessions; otherwise, each learner follows the same learning process at the same time. After this initial *learning phase*, the simulated learners are left without reviewing anything during the *retention phase*. At the end of this final period, the simulation stops. Average Correctness Probability (*ACP*) is a learner performance measure that is recorded throughout the learning and retention phases and that is used to compare KC selection strategies. Algorithm 1 finally returns the metrics ACP_L (Average Correctness Probability during Learning) and ACP_R (Average Correctness Probability during Retention) for the simulated learner. Details about the computation of these metrics are given in Subsection 4.3.

The complete simulation protocol is described in Algorithm 2. In order to measure the robustness of our results and to take into account a wide variety of learning scenarios and forgetting behaviors, we run N times the simulation procedure, each time with a different randomly generated group of learners, items, KCs, and q-matrix. In this way, we can also compute measures of variability from one simulation run to another. To compare our adaptive spacing heuristics, each learner within a given simulation is assigned to each spacing strategy. More specifically, in a simulation run, each learner follows the learning and retention process described in Algorithm 1 for each (KC selection strategy, item selection strategy) pair. Figure 2 schematizes this complete process. Of course, when a learner starts a new learning process, their previous practice history is completely reset. At the end, Algorithm 2 returns the performance metrics of each strategy, averaged over all simulation runs and all learners.

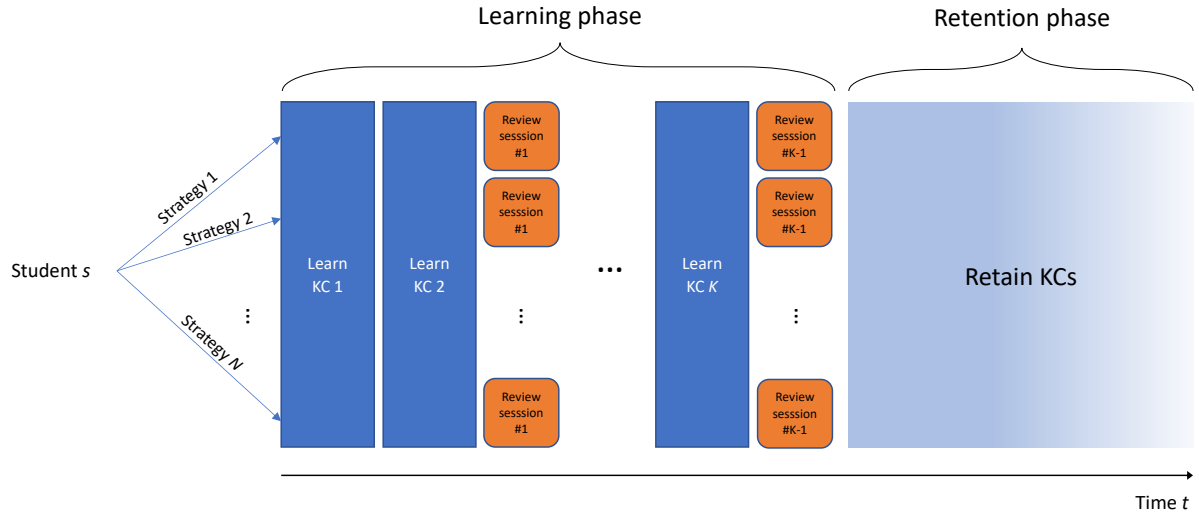


Figure 2: Learning and memorization process for a student s in a run i

4.2. Simulation Parameters

4.2.1. Generated Parameters

DAS3H Parameters Generation. To simulate learning trajectories with our DAS3H model, we choose to randomly and independently sample learner (α_s), item (δ_j), KC (β_k), and learning/forgetting ($\theta_{k,w}$) parameters from specified probability distributions. Randomly generating these parameters from fixed probability distributions allows us to control the characteristics of our simulations, and in particular, the learning and forgetting dynamics. It also allows us to take into account a wide variety of learning and forgetting behaviors.

In the past, other research works have also chosen to compare their adaptive spacing algorithms on simulated data, using forgetting models and randomly sampled (Reddy et al., 2017; Yang et al., 2020) or arbitrarily fixed (Khajah et al., 2014) parameters.

Table 1 details the probability distributions that we choose. In terms of learning and memory, our choice for α_s, δ_j and β_k means that without any practice, an average student will have very few chances (around 0.12) to solve an average item involving a single average KC. If they manage to solve it though, it could come from a very easy item or KC, or prior knowledge from the student. As the number of additional KC labels increases, the correctness probability may decrease or increase as we add β_k to Equation 3, depending on the sign of each individual β_k.

Table 1: Distributional assumptions of our DAS3H parameters

Parameter	Distribution
α_s	$\mathcal{N}(0,1)$
δ_j	$\mathcal{N}(1,1)$
β_k	$\mathcal{N}(-1,1)$
$\theta_{k,2w+1}, \theta_{k,2w+2}$	$\mathcal{U}_{[0,2]}$

We chose the $\theta_{k,w}$ probability distribution such that failing to solve an item that involves a KC would still improve future correctness probability on that KC but less than correctly solving the item: this means $\theta_{k,2w+1} > 0$ and $\theta_{k,2w+2} > 0$. Previous work also used this type of forgetting behaviors (Reddy et al., 2017; Upadhyay et al., 2018; Tabibian et al., 2019). We wanted to generate diverse learning and forgetting curves, exhibiting a wide range of behaviors, so we therefore chose a uniform $\mathcal{U}_{[0,2]}$ law for these parameters.

Q-MATRICES GENERATION It was also necessary to generate random q-matrices for determining item-KC relationships. First of all, let us remind that the index of a KC corresponds to its order of introduction within the curriculum. We generated for each KC k the same amount of practice items: then, we added a random amount of other KC labels to each of these items. Thus, a practice item for KC k necessarily involves KC k but can also involve other (previously learned) KCs. More precisely, the amount of new KC labels for an item initially associated with KC k is each time randomly generated following $\mathcal{U}_{[\![0,\min(k-1,2)]\!]}$. A randomly generated (according to our protocol) q-matrix is plotted in Figure 3. Items in this q-matrix have an average of 1.84 KC labels.

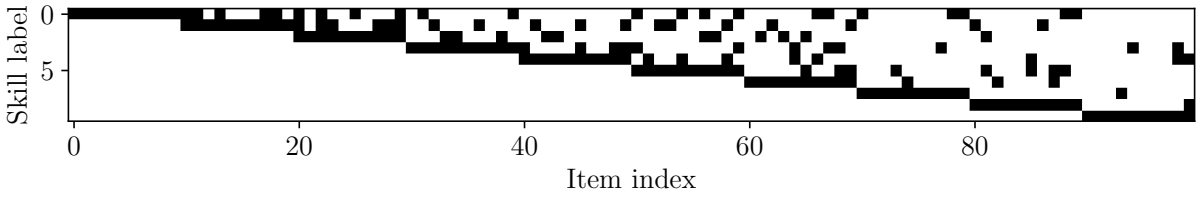


Figure 3: Random q-matrix (10 KCs, 10 practice items per KC). Each black square indicates that the item from the column involves the KC from the row. Notice that practice items cannot include KCs after the current week's KC, hence the upper-triangular matrix form. For presentation purposes, the q-matrix is transposed here.

This procedure generates realistic q-matrices because :

- It mimics a teaching program and its sequence of introduction of the different KCs;
- Each KC appears a minimum number of times in a q-matrix, and an item can have at most 3 KC labels;
- If one wants to review KC k and it has already been introduced, it is always possible to find an item that involves k and *only* previous KCs.

4.2.2. Fixed Parameters

In our experiments, we set:

- N the number of simulation runs to 100;
- S the number of simulated students per run to 500;
- K the number of weeks in the learning phase (and the number of KCs to learn) to 10;

- The number of weeks in the retention phase to 6;
- r the number of items to select per review session to 3;
- c the number of generated practice items per KC to 20;
- The maximum number of additional KC labels per item to 2.

4.3. Student Performance Metrics

We describe in this Subsection the metrics in which we were interested for measuring student long-term memory retention and comparing the strategies' efficiencies.

We choose to measure KC mastery of student s at time t by their ability to answer an average item involving that KC, i.e. by their ability to generalize to unseen material. We were thus interested in measuring and comparing the *Average Correctness Probability* (*ACP*) over all KCs of our synthetic students. This choice was inspired by previous works in the literature (Hunziker et al., 2019; Kang et al., 2014; Yang et al., 2020). We give below the *ACP* formula for a single student[11] s, at time t:

$$ACP_{s,t} = \frac{1}{K} \sum_{k=1}^{K} \mathbb{P}\left(Y_{s,k,t} = 1\right) \tag{5}$$

Since DAS3H does not directly output the correctness probability for a given KC, we compute the *ACP* by replacing the item difficulty δ_j by the mean of the probability distribution used to sample the difficulties of the items (in our case, 1) in the model. This is the equivalent of the idealized learning curves from Goutte et al. (2018). We use this *ACP* to compare different heuristics according to the following definitions.

Definition 1 – *Average Correctness Probability* (*ACP*) of a heuristic. The *ACP* of a heuristic h, denoted ACP_t^h, is equal to the average of the individual *ACP* scores of the learners assigned to this heuristic, over all simulation runs.

$$ACP_t^h = \frac{1}{|S_h| \times |N|} \sum_{n \in N} \sum_{s \in S_h} ACP_{s_n,t} \tag{6}$$

with :

- S_h the set of learners associated with the heuristic h, and $|S_h|$ its cardinality;
- N the set of simulations and $|N|$ its cardinality;

However, and contrary to other previous works (Khajah et al., 2014), we argue that evaluating delayed student performance using only a few point estimates does not adequately match our goal of optimizing long-term memory retention. Thus, we introduce our two final student performance metrics.

[11] Let us remind that every synthetic student is assigned to all strategies; this *ACP* metric is thus computed for every student, *for every spacing strategy*.

Definition 2 – ACP_L of a heuristic. The ACP_L^h measures the performance of the learners assigned to a heuristic h during the learning period :

$$ACP_L^h = \frac{1}{10}\sum_{t=1}^{10} ACP_t^h \tag{7}$$

Definition 3 – ACP_R of a heuristic. The ACP_R^h measures the performance of the learners assigned to a heuristic h during the retention period:

$$ACP_R^h = \frac{1}{6}\sum_{t=11}^{16} ACP_t^h \tag{8}$$

In what follows, unless otherwise stated, we remove the index h from these metrics. Intuitively, these ACP_L and ACP_R scores measure, respectively, how well a student s is able to apply the set of KCs during the whole learning and retention periods. Since the timestamps t are discrete, this score is only an estimate of the true average correctness probability over these periods. The ACP_R formalizes the objective criterion that we defined in Section 1.

During the learning phase, to collect student performance measures, the *ACP* was computed right after the last reviewing item of the week was presented to the student[12]. During the retention phase, we computed the *ACP* every week between the end of the learning phase and the maximum retention delay that we specified in the algorithm. In our simulations, this delay was equal to 6 weeks. The ACP_R is more interesting to us than the ACP_L since it measures retention after the last reviewing session; however, we found the analysis of the ACP_L interesting too because it synthesizes the heuristic's performance *during* learning.

5. Selection Strategies

Selecting practice items is trickier in our case than for memorizing single pieces of knowledge, such as vocabulary in a foreign language. Indeed, we want to select the best item for optimizing long-term mastery of a set of underlying KCs: the selection step is thus twofold. First, we need to select the KC or subset of KCs for the learner, and then we must select an item among the items that involve the KC or the KC subset.

Notice that this formalization encompasses the traditional adaptive spacing framework: it only requires associating every item with a distinct KC. This wipes out the need to select an item after the KC.

In this Section, we first present the different KC selection heuristics that we developed, implemented, and compared in our experiments. In particular, we propose a new greedy KC subset selection procedure. Finally, we describe the item selection strategy that we considered in our experiments.

In what follows, *feasible* KCs and items respectively refer to the set of KCs that have been seen previously by the student (excluding the current week's KC) and the set of items that involve the chosen KC (or KCs) and *only* feasible KCs.

[12]So, during the learning phase, our metric involves in its computation the correctness probability for KCs that have not been seen yet by the simulated student.

5.1. KC Selection Strategies

The Select skill procedure in Algorithm 1 selects the KC or subset of KCs to review at a given time t. We remind our reader that in our simulations, KC i is learned for the first time by any student at week i. In what follows, $k^\star$ gives the index of the KC that is chosen by the heuristic.

5.1.1. Single KC Selection

Baseline No review.

Random Choose the KC to review uniformly at random from the subset of feasible KCs:

$$\text{Sample KC } k^\star \text{ from } \mathcal{U}_{[\![1,k_{current}-1]\!]} \tag{9}$$

with $k_{current}$ the KC that was introduced on the current week. $k_{current} \geq 2$ because students begin to review at week 2.

μ-back Choose the KC that was learned μ weeks ago – i.e., μ KCs backward. Its parameter μ can take strictly positive integer values. It is inspired from the cognitive science literature and from Khajah et al. (2014). This non-adaptive strategy yields a fixed reviewing schedule, that is shared by all students with the same μ parameter and that does not adapt to students' performance. But its strength lies in its simplicity and in the fact that it does not need to be backed by a student predictive model. It is also directly operational for human teachers.

$$\text{Select KC } k^\star = \max\left(k_{current} - \mu, 1\right) \tag{10}$$

θ-threshold Select the KC whose current correctness probability is closest to a fixed value $\theta \in [0, 1]$. This strategy is also inspired from the cognitive science literature and from Khajah et al. (2014). Intuitively, this strategy seeks to choose the KC that is on the verge of being forgotten: it is thus consistent with Bjork's notion of "desirable difficulties" (Bjork, 1994). This strategy requires a student predictive model that is able to output the current probability that the student can correctly answer an average difficulty item involving a single KC k.

$$\text{Select KC } k^\star \text{ s.t. } k^\star \in \underset{k \in [\![1,k_{current}-1]\!]}{\operatorname{argmin}} \left|\mathbb{P}\left(Y_{s,k,t_{current}} = 1\right) - \theta\right| \tag{11}$$

with $t_{current}$ the current timestamp.

Greedy This strategy is adapted to our KC selection problem from Hunziker et al. (2019). It consists of selecting the KC k for which the expected student performance gain, averaged over all future retention timestamps and all KCs (i.e., after the last reviewing session has been done), after choosing k, is highest. Like *θ-threshold*, this strategy also requires a student predictive model, but here the model's predictions extend beyond the current week. Thus, the model must be particularly reliable since the objective function to optimize includes model inferences on future correctness for several future timestamps. On the other hand, contrary to *θ-threshold*, one does not need to select a hyperparameter for this KC selection strategy. Since we want to optimize long-term memory, we choose to select a time horizon different from Hunziker et al.: instead of optimizing future *ACP* over all timestamps, we focus on the retention period.

$$\text{Select KC } k^\star \text{ s.t. } k^\star \in \underset{k \in [\![1, k_{current}-1]\!]}{\text{argmax}} U\left(k | \sigma_{1:t_{current}-1}, y_{1:t_{current}-1}, \tau_0\right) \tag{12}$$

with:

$$U\left(k|\sigma_{1:t-1}, y_{1:t-1}, \tau_0\right) = \mathbb{E}_{y_t}\left[f\left(\sigma_{1:t-1} \oplus k, y_{1:t-1} \oplus y_t; \tau_0\right) - f\left(\sigma_{1:t-1}, y_{1:t-1}; \tau_0\right)\right] \tag{13}$$

$$f\left(\sigma_{1:t}, y_{1:t}; \tau_0\right) = \frac{1}{K\left(T - \tau_0 + 1\right)} \sum_{k=1}^{K} \sum_{\tau=\tau_0}^{T} \mathbb{P}\left(Y_{s,k,\tau} = 1 | \sigma_{1:t}, y_{1:t}\right) \tag{14}$$

and:

- $\sigma_{1:t}$ the sequence of encountered KCs up to t (included);
- $y_{1:t}$ the outcomes of these interactions;
- τ_0 the first timestamp of the retention period;
- T the last timestamp of the retention period;
- K the total number of KCs.

The expectation is taken over the (unknown) outcome of student s answering an average item only labeled with KC k at time t. The $\oplus$ symbol represents the concatenation operation. Intuitively, f represents the average correctness probability over all KCs and all future *retention* timestamps, given the past study history of student s: $(\sigma_{1:t}, y_{1:t})$.

But since we assumed that our KCs are independent from each other, practicing a KC k has only an impact on the future correctness probability of *this* KC k. Moreover, our KCs are equally weighted in the objective function U. Thus, we can replace our function f by a simpler and less computationally expensive one, in which we removed the second sum over the KCs:

$$f^\star\left(\sigma_{1:t}, y_{1:t}; k, \tau_0\right) = \frac{1}{T - \tau_0 + 1} \sum_{\tau=\tau_0}^{T} \mathbb{P}\left(Y_{s,k,\tau} = 1 | \sigma_{1:t}, y_{1:t}\right) \tag{15}$$

Table 2 synthesizes the parameter domains for every KC selection strategy that we developed and compared.

Two KC selection heuristics need to have access to some information about the probability that a student will correctly answer an item involving a given KC now or in the future: *θ-threshold* and *Greedy*. Thus, they would require in real life having data of past interactions with the set of practice items, and maybe even from the current students. This is one of the disadvantages of these heuristics: they need a reliable student predictive model for scheduling reviews. This model should be periodically trained on new data generated from the students. This also makes these two heuristics more complex to use and implement. Following the experimental protocol in Khajah et al. (2014), we chose to use the DAS3H model to get these probability estimates in our own experiments. Notice, however, that the *Greedy* and *θ-threshold* heuristics are model-agnostic and could be straightforwardly used with any other student model, provided that this model can output accurate predictions for KC correctness probabilities.

Table 2: Parameter domains for the different KC selection heuristics

KC selection heuristic	Parameter domain
Baseline	$\varnothing$
Random	$\varnothing$
μ-*back*	$\mathbb{N}^{*+}$
θ-*threshold*	$[0, 1]$
Greedy	$\varnothing$

Finally, we chose not to compare our KC selection strategies to the Leitner system (Leitner, 1972; Reddy et al., 2016) because we did not manage to extend it successfully to items involving multiple KCs. Also, we did not compare our KC selection heuristics to SuperMemo (Wozniak and Gorzelanczyk, 1994) because it was not adapted to our experimental setting with fixed review sessions one week apart and to our student simulator (which outputs correctness probabilities).

5.1.2. KC Subset Selection

One of the main contributions of this article is a new greedy[13] procedure for selecting the most promising subset of KCs to review, rather than the single most promising KC. This procedure is generic and can be adapted to any selection strategy, as long as it can provide a review priority ranking of the KCs at any given time t. For example, it is not compatible with μ-*back* since μ-*back* cannot output such a ranking.

One of the main differences with previous work is that in our research setting, items potentially involve multiple KCs at the same time. We assume that making the student practice an item that involves multiple KCs at the same time will improve the student's mastery in *each* of these KCs. This assumption led us to develop our greedy KC subset selection procedure, described in Algorithm 3.

Algorithm 3 Greedy KC Subset Selection Procedure

procedure SELECT MULTI-KCs(KC selection strategy, current week k, q-matrix Q)
  $\sigma \leftarrow$ RANKKCs(KC selection strategy, k) ▷ Ranks *feasible* KCs
  $chosenKCs \leftarrow \{\}$
  for σ_i in σ **do**
    $\Lambda \leftarrow$ GETITEMSLIST(Q, $chosenKCs \cup \{\sigma_i\}$, k)
    if $\Lambda \neq \{\}$ **then**
      $chosenKCs \leftarrow chosenKCs \cup \{\sigma_i\}$
  $\Lambda \leftarrow$ GETITEMSLIST(Q, $chosenKCs$, k)
  $j^\star \leftarrow \mathcal{U}_\Lambda$
  return $j^\star$
end procedure

[13]Note that this "greedy" procedure is quite different from the heuristic called *Greedy*. *Greedy* selects a KC to review at time t whereas this greedy procedure selects, from a compatible single KC selection heuristic, the optimal subset of KCs to review at time t.

Table 3: Table comparing the characteristics of the different KC selection heuristics that we used in our adaptive spacing algorithms. The "Temporal horizon" column gives the temporal horizon considered by the heuristic for its decision. The higher the number of "+" in the "Implementation difficulty" column, the more difficult is the heuristic to implement.

Heuristic	Model	Adaptive	Hyperparameter	Multi-KC	Temporal horizon	Implementation difficulty
Random	Without	No	Without	No	None	+
μ-back	Without	No	With	No	None	+
θ-threshold	With	Yes	With	No	Short	++
θ-threshold (multi.)	With	Yes	With	Yes	Short	+++
Greedy	With	Yes	Without	No	Long	+++
Greedy (multi.)	With	Yes	Without	Yes	Long	++++

First, the procedure calls the RankKCs routine, which ranks the feasible KCs indices by decreasing order of review priority, according to the chosen KC selection strategy. Then, the procedure loops over the KC indices of the σ ranking from the most urgent to the least urgent KC. If there exist feasible items that involve all the already selected KCs **and** the considered KC, then this KC is added to the list of selected KCs; otherwise it is not. For this purpose, GetItemsList(Q, L, k) returns the list of feasible items from q-matrice Q that involve all KCs from list L and no KC after k. The selection ends when the list of KCs has been looped over.

Finally, the algorithm chooses uniformly at random an item from the list of feasible items that involve all the chosen KCs[14]: unlike single KC selection procedures, this procedure therefore returns an item, not a KC. This is a greedy selection criterion. Items selected with these strategies generally involve at least two KCs, but they may not involve more than two KCs (if there is no item available that meets all the criteria). Depending on the generated q-matrix and the selected item, single KC selection strategies can also make a student review multiple KCs with a single item, but the other KCs are then randomly selected. Note however that in our simulations, the number of items to select for a student is identical for every spacing algorithm.

Thus, we propose two "multi-KC" versions of our single KC selection heuristics *θ-threshold* and *Greedy*.

θ-threshold (multi.) Uses the Select Multi-KCs procedure with the RankKCs routine which ranks the KCs by increasing order of distance (in absolute value) to $\theta \in [0, 1]$.

Greedy (multi.) Uses the Select Multi-KCs procedure with the RankKCs routine which ranks the KCs by decreasing order of expected gain $f^\star$ (see Equation 15). Since the KCs are independent of each other, we can iteratively select each new KC to be added to the selected subset; thanks to this property, we do not have to recompute the ranking each time we add a KC to the set of selected KCs.

We synthesized the different KC selection heuristics that we used in our experiments in Table 3. We also indicate the main characteristics of these algorithms in this table.

Finally, we used in our experiments different grids of hyperparameter values for the three heuristics *θ-threshold*, *θ-threshold (multi.)* and *μ-back*:

[14]This list cannot be empty because the first KC of the ranking is necessarily added and the procedure adds another KC only if there exist items that involve all the chosen KCs.

- *θ-threshold*, *θ-threshold (multi.)*: 11 evenly spaced θ that split $[0, 1]$ into 10 equal size intervals;
- *μ-back*: {1,2,3}.

In our results, unless otherwise stated, we only indicate the performance of the optimal hyperparameter, i.e., the one that optimizes the ACP_R.

5.2. Item Selection Strategy

The Select item procedure of Algorithm 1 selects, from a KC or a set of KCs, the item to be presented to a learner.

After selecting the KC or the set of KCs, we must then choose the specific item to present to student s. We proposed and studied the *Random KC* item selection strategy, which selects an item uniformly at random among the set of feasible items.

6. Results

In this Section, we describe the results of our synthetic experiments comparing adaptive spacing heuristics for optimizing long-term KC mastery on simulated students. Python code to reproduce our results has been published on GitHub[15].

Our main results, following the experimental protocol described in Section 4, are presented in Tables 4 and 5. We computed the ACP_L and ACP_R scores for each learning trajectory, and we averaged the performance metrics of each KC selection strategy to get these results. Standard errors of the mean across all simulation runs are also reported. For *μ-back* and *θ-threshold*, we present the results only for the μ and θ parameters that maximize the ACP_R metric, and the column *Best parameter* gives this parameter. Further analyses on the impact of these parameters on the heuristics' performance are given in Subsection 7.2.

In order to enable a more readable comparison of the performances of the heuristics, we report in Table 5 the performances of each KC selection strategy, relatively to the *Random* selection strategy. Let us call :

- $ACP^h_{s,t,i}$ the *ACP* score of learner s at time t with the strategy h in the run i;
- $ACP^{random}_{s,t,i}$ the *ACP* score of learner s at time t with the strategy *Random* in the run i;
- $\rho^h_{s,t,i}$ the *relative ACP* score of the learner s at time t with the strategy h in the run i.

Thus we define:

$$\rho^h_{s,t,i} = \frac{ACP^h_{s,t,i} - ACP^{random}_{s,t,i}}{ACP^{random}_{s,t,i}} \tag{16}$$

The relative score ρ thus expresses how much better (or worse) the performance of the strategy h is compared to the *Random* strategy, as a percentage of the *ACP* of *Random*. In what follows, we do not show the results of the *Random* selection strategy in these tables, because

[15] https://github.com/BenoitChoffin/multiskill_adaptive_spacing

Table 4: ACP_L and ACP_R comparison between all KC selection strategies. Results are averaged over all runs, KCs, students. Standard errors of the mean across the runs are reported. A higher *ACP* is better.

KC strategy	Best parameter	ACP_L	ACP_R
No review	∅	0.318 ± 0.001	0.297 ± 0.001
Random	∅	0.567 ± 0.000	0.695 ± 0.001
μ-back	1	0.589 ± 0.000	0.753 ± 0.001
θ-threshold	0.4	0.599 ± 0.000	0.754 ± 0.001
θ-threshold (multi.)	0.2	0.604 ± 0.000	0.814 ± 0.000
Greedy	∅	0.579 ± 0.000	0.759 ± 0.001
Greedy (multi.)	∅	0.592 ± 0.000	0.816 ± 0.000

Table 5: Relative ACP_L and ACP_R comparison between all KC selection strategies. Results are averaged over all runs, KCs, students. Standard errors of the mean across all runs are reported. A higher *ACP* is better.

KC strategy	Best parameter	Relative ACP_L	Relative ACP_R
No review	∅	−0.404 ± 0.001	−0.593 ± 0.001
μ-back	1	0.036 ± 0.000	0.093 ± 0.000
θ-threshold	0.4	0.050 ± 0.000	0.089 ± 0.000
θ-threshold (multi.)	0.2	0.059 ± 0.000	0.189 ± 0.001
Greedy	∅	0.017 ± 0.000	0.100 ± 0.000
Greedy (multi.)	∅	0.039 ± 0.000	0.194 ± 0.000

they are by definition equal to 0. Then, as with the *ACP* scores, the relative scores are averaged by period (learning and retention) between learners and simulation runs. It should be noted that each relative *ACP* score is computed *during* the simulations and then averaged afterwards: this explains why the computation of this metric from Table 4 does not yield the exact same results as in Table 5.

6.1. Description of the Results

As expected, the *No review* strategy is outperformed by every KC selection strategy, both during learning and retention periods. This is due to the low amount of times that simulated students assigned to this strategy have seen each KC (one per week, which yields a total of 10 KC attempts). It is also the only one to have an ACP_L metric higher than the ACP_R. This is because the computation of the ACP_L is averaged over all weeks: thus, the very first weeks of the learning period have a low score because the student has not mastered all KCs. During retention, on the contrary, all KCs have been seen at least once by every student.

θ-threshold is the best performing single KC selection strategy during learning with an ACP_L of .599 on average; *μ-back* achieves .589 of ACP_L, outperforming *Greedy* during learning. Indeed, *Greedy* locally optimizes the average correctness probability over all KCs during retention and does not aim at optimizing correctness probability during learning, which may explain these

results. Also, *θ-threshold* selects the KC whose immediate correctness probability is closest to a fixed value θ (with here $\theta^\star = .4$): this could explain why it performs that well during learning.

However, during the retention phase, this pattern changes: *θ-threshold* performs on-par with *μ-back*, whereas *Greedy* only slightly outperforms the two other strategies. We hypothesize that the poor performance of *Greedy* comes from the fact that it only considers the direct impact of choosing a KC, without taking the whole sequence of reviews into account.

During the learning period, *θ-threshold (multi.)* outperforms all other strategies. *Greedy (multi.)* only achieves an ACP_L of .592, which is lower than vanilla *θ-threshold* and only slightly better than *μ-back*. These multi-KC results are consistent with the results of their single KC counterparts. On the other hand, both multi-KC algorithms substantially outperform all the other heuristics in terms of ACP_R with respectively .814 and .816 in average for *θ-threshold (multi.)* and *Greedy (multi.)*. These results are evidence that selecting the best subset of KCs improves over selecting the single best KC at a given timestamp.

6.2. Testing the Significance of Performance Gaps between Strategies

To test whether these results are significantly different from one KC selection strategy to another, we performed a series of statistical tests: for each pair of KC selection strategies, we tested with the help of a two-sided sign test if the two KC selection strategies produced statistically different student performances.

We used sign tests because :

1. It is a simple test that requires only few assumptions: mutual independence of the pairs, ordinal scale of measurement within each pair;

2. We couldn't verify that assumptions for similar tests but with more statistical power, such as the matched Student test (hypothesis of normality of the distribution of differences within each pair), held.

All strategies' performances were statistically different from each other at the .001 level.

6.3. Temporal Evolution of the *ACP* Scores

To visualize our results differently, Figure 4 displays the temporal evolution of the *ACP* score, averaged over all runs and students, and broken down by KC selection strategy. Figure 5 plots the same evolution but with the relative *ACP* metric (same as in Table 5) instead. Again, we only display results for the best parameter found for *μ-back* and *θ-threshold* strategies. To ensure maximal readability of the plots and since we already reported variability measures and significance tests for these results before, we purposely do not report confidence intervals by timestamps in these figures.

We see that without any review (*No review* strategy), simulated students only slightly increase their *ACP* during the learning period (thanks to the +1 attempt added to their learning history at each week). During retention, and like every other strategy, their *ACP* score decreases. For all other strategies, all *ACP* curves are merged up to week 2 and progressively differentiate afterwards. This is explained by the fact that there is only one available choice at week 2: reviewing an item that only involves KC 1.

During learning, the *θ-threshold* strategies outperform all other strategies including *Greedy (multi.)*, confirming our findings from Subsection 6.1 (Table 4). As soon as the retention period

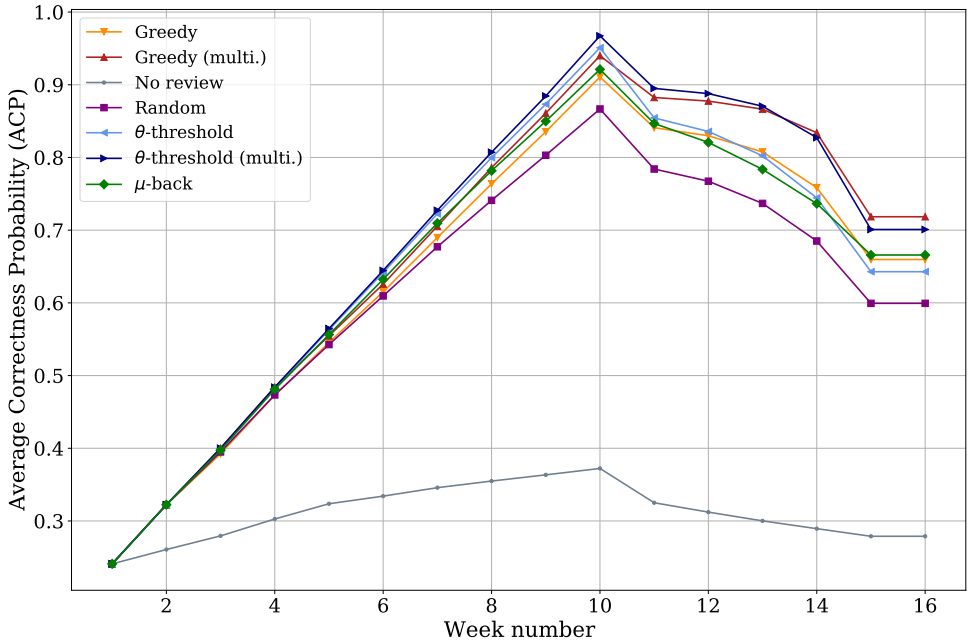


Figure 4: Comparison of the *ACP* score evolution over time between different KC selection strategies. For the parameterized strategies, only the best performing parameter is displayed. A higher *ACP* is better.

begins, *θ-threshold* starts to drastically decrease and ends up as the worst performing strategy (if we except *Random* and *No review*). *μ-back* has the most consistent *ACP* decay rate; *Greedy* outperforms it during most of the retention period, before ending at a similar *ACP*. The multi-KC heuristics are both the best performing algorithms during the whole retention period. At first, *θ-threshold (multi.)* slightly outperforms *Greedy (multi.)* but ends up at a lower *ACP*. These combined results suggest that *Greedy* might be more suited for long-term memorization, whereas *θ-threshold* focuses more on short- and medium-term mastery.

6.4. Synthesis of the Results

To sum up this section, we compared different KC selection strategies for adaptive spacing algorithms, including heuristics that selected the final set of KCs at once. We showed that *θ-threshold* and *μ-back* perform similarly during retention but we found also that *θ-threshold* improved student performance during learning as well. However, the *Greedy* strategy only slightly improved over the other competing strategies. We hypothesized that this comes from the local optimization procedure, unable to take the whole sequence of future reviews into account. Finally, we showed that the multi-KC versions of two heuristics outperform all other single KC selection heuristics.

7. Discussion

In this Section, we extend the analysis of the results presented in the previous section. More precisely, we wish to evaluate the robustness of our algorithms under different conditions (Subsection 7.1) but also to better understand their performance (Subsection 7.2). We finally discuss

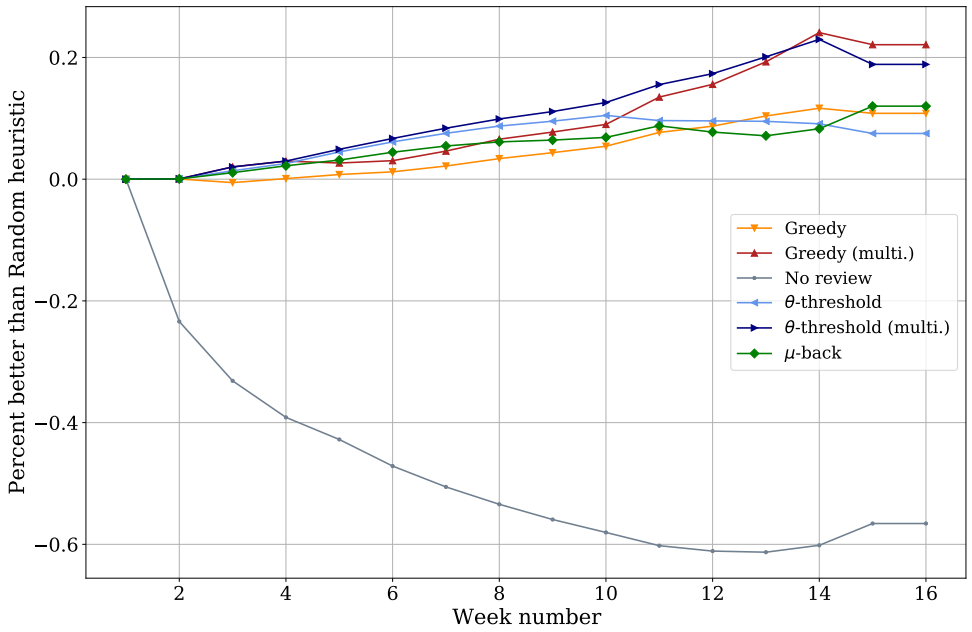


Figure 5: Comparison of the *relative ACP* score evolution over time between different KC selection strategies. For the parameterized strategies, only the best performing parameter is displayed. A higher *ACP* is better.

a few challenges that a real-world implementation of our algorithms would pose (Subsection 7.3).

7.1. Is the Performance of the Spacing Algorithms Sensitive to Atypical Forgetting Behaviors?

7.1.1. Motivation

In our main experiment, we independently sampled the DAS3H $\theta_{k,w}$ parameters from a $\mathcal{U}_{[0,2]}$. This choice determined a specific forgetting behavior: successfully solving an item involving a given KC would improve future correctness probability on this KC to a greater extent than failing at this item. However, other probability distributions and forgetting behaviors might also be plausible and realistic. For instance, sampling the $\theta_{k,w}$ parameters such that:

$$\forall k, w \in [\![1, K]\!] \times [\![1, W]\!], \theta_{k,2w+1} < 0 < \theta_{k,2w+2} \text{ and } |\theta_{k,2w+1}| < |\theta_{k,2w+2}|. \tag{17}$$

This would mean that succeeding at solving an item that involves a given KC would still improve future correctness probability on this KC but less than failing at this item. An example of such forgetting curves are represented in Figure 6.

This situation might seem counterintuitive. After all, a good answer is a priori a stronger signal of KC mastery than a wrong answer. A possible interpretation of such a situation comes from findings from the cognitive science literature (Rowland, 2014). In this meta-analysis, Rowland shows that high initial student performance can in some cases be associated with a smaller testing effect than for low initial student performance. This is in particular the case when feedback is available after each student's answer. Otherwise, in the absence of such

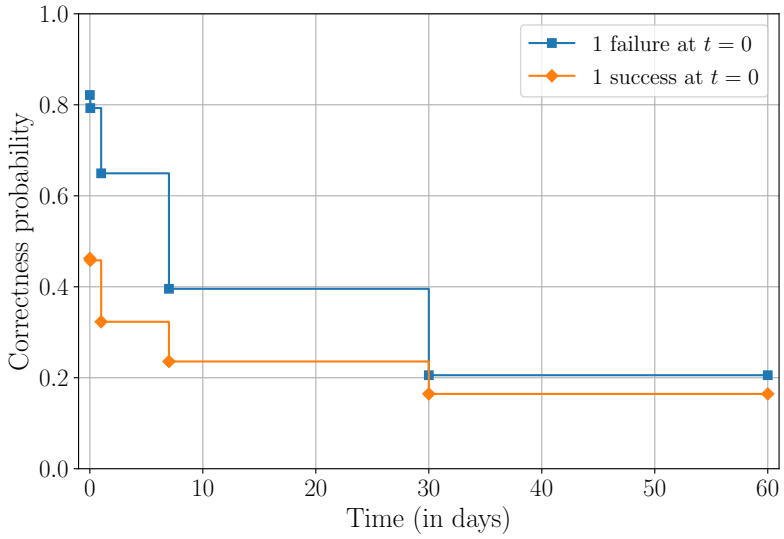


Figure 6: Example forgetting curves generated from our DAS3H simulator. Two situations are presented: in blue, an average proficiency student practices the KC at $t = 0$ but fails; in orange, they succeed. These curves are similar to the idealized learning curves from Goutte et al. (2018).

feedback, correct answers tend to increase future recall probability more than wrong answers. One could also imagine that a wrong answer encourages the learner to understand their mistake and therefore generates a greater increase of the future correctness probability than a correct answer. This setting has, to the best of our knowledge, not been previously used in the adaptive spacing literature (Hunziker et al., 2019; Tabibian et al., 2019; Reddy et al., 2017). With this additional experience, we want to test the robustness of our KC selection strategies on alternative but realistic learning and forgetting behaviors. The one we are using here is an example of such atypical situations.

7.1.2. Results

As a consequence, we performed the same experiments as in Section 6 but we changed the sampling distributions for the $\theta_{k,w}$. More precisely, we first generate the $\theta_{k,2w+2}$ from a $\mathcal{U}_{[0,2]}$ and then

$$\forall k, w \in [\![1, K]\!] \times [\![1, W]\!], \theta_{k,2w+1} \sim \mathcal{U}_{[-\theta_{k,2w+2},0]}. \tag{18}$$

This forces the odd coefficients (associated with the successes) to be negative but lower in absolute value than the even coefficients (associated with the attempts). Apart from this simple difference, the whole experimental protocol remains the same.

Results are given in Tables 6 and 7. We performed pairwise difference significance tests between all KC selection strategies as in Section 6 and all pairwise differences were significant at the .001 level. We can see that overall, the performance of all KC selection heuristics decreases substantially, compared to the results in Section 6. Also, all ACP_L scores are now higher than the ACP_R scores, meaning that forgetting imposes a greater burden on student's long-term mastery.

Here, μ-*back* performs more poorly than *Random* and its best parameter is now $\mu = 2$. We believe that μ-*back*'s underperformance comes from the fact that choosing recent KCs increases

Table 6: ACP_L and ACP_R comparison between all KC selection strategies, under an atypical forgetting behavior scenario. Results are averaged over all runs, KCs, students. Standard errors of the mean are reported. A higher *ACP* is better.

KC strategy	Best parameter	ACP_L	ACP_R
No review	∅	0.316 ± 0.001	0.304 ± 0.001
Random	∅	0.465 ± 0.001	0.418 ± 0.001
μ-back	2	0.453 ± 0.001	0.415 ± 0.001
θ-threshold	0.5	0.468 ± 0.001	0.419 ± 0.001
θ-threshold (multi.)	0.2	0.505 ± 0.001	0.434 ± 0.001
Greedy	∅	0.470 ± 0.001	0.441 ± 0.001
Greedy (multi.)	∅	0.482 ± 0.001	0.458 ± 0.001

Table 7: Relative ACP_L and ACP_R comparison between all KC selection strategies, under an atypical forgetting behavior scenario. Results are averaged over all runs, KCs, students. Standard errors of the mean are reported. A higher *ACP* is better.

KC strategy	Best parameter	Relative ACP_L	Relative ACP_R
No review	∅	−0.318 ± 0.001	−0.303 ± 0.001
μ-back	2	−0.021 ± 0.000	−0.007 ± 0.000
θ-threshold	0.5	0.007 ± 0.000	0.005 ± 0.000
θ-threshold (multi.)	0.2	0.083 ± 0.000	0.040 ± 0.000
Greedy	∅	0.012 ± 0.000	0.073 ± 0.000
Greedy (multi.)	∅	0.035 ± 0.000	0.112 ± 0.000

the probability of student correctness during review. Since, in this alternative setting, failures have a more positive impact on future correctness probability than successes, this could explain why *μ-back* performs more poorly than *Random*. *θ-threshold* also suffers from this setting: even though it outperforms *μ-back*, it barely manages to improve over the *Random* baseline. Interestingly, *Greedy* manages to adapt to this different situation and substantially outperforms all other single KC strategies during retention while performing on-par with *Random* and *θ-threshold* during learning. *Greedy (multi.)* too outperforms all other KC selection strategies during both periods – except *θ-threshold (multi.)* during learning. We hypothesize that this comes from the increased flexibility of our proposed *Greedy* algorithms that allows them to adapt to diverse learning scenarios.

To conclude this subsection, our results suggest that among our KC selection heuristics, *Greedy* and *Greedy (multi.)* are the most appropriate algorithms for dealing with atypical learning and forgetting behaviors.

7.2. What is the Impact of the Parameter Choice on Student Performance for the θ-*threshold* and μ-*back* Strategies?

7.2.1. Motivation

Khajah et al. (2014) compare the relative performance of a grid of θ and μ parameters for θ-*threshold* and μ-*back* on both of their memory models. Relative performance is computed relatively to the performance obtained by the best reviewing schedule.

Our student simulator model (DAS3H) is an extension of the DASH model (Lindsey et al., 2014). DASH's memory module h_θ itself was inspired by ACT-R and MCM – both memory models which were used by Khajah et al.. We also use two KC selection heuristics (θ-*threshold* and μ-*back*) inspired by Khajah et al. (2014) and proposed an extension of θ-*threshold* to select an item with *multiple* promising KCs.

Considering these similarities (models and strategies) and differences (items are labeled with multiple KCs in our setting), we wanted to know if our result patterns match with those of Khajah et al.. Moreover, this problem is of essential practical importance: since it would be costly to test on real learners a grid of parameters for θ-*threshold* and μ-*back*, it is preferable to analyze them beforehand with simulations (Khajah et al., 2014; Lindsey et al., 2014).

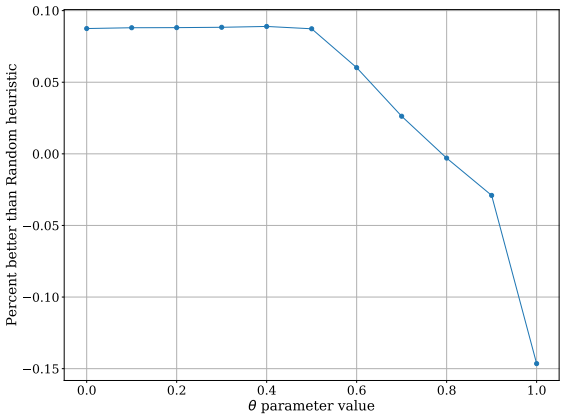
7.2.2. Results

θ-threshold Thus, we compared the ACP_R scores for different θ parameters, for θ-*threshold* and θ-*threshold (multi.)*. Results, relative to the *Random* heuristic, are plotted on the left-hand side of Figures 7 and 8. For this analysis, we used the data from the experiments described in Section 6. To better understand these results, we also compare in the right-hand side of each of these figures the ACP_R scores broken down by θ parameter and by KC introduction index.

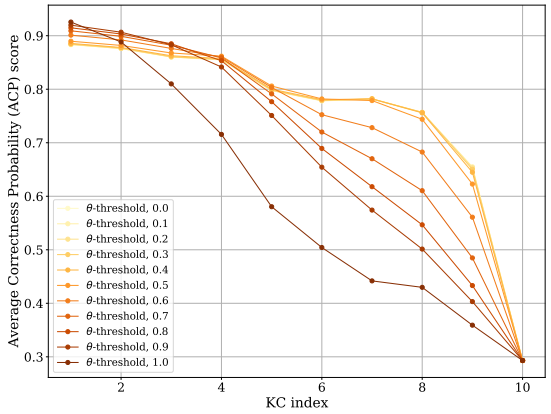
Notice that this comparison with Khajah et al.'s work should also be taken with caution since they were able to compute the performance yielded by the best reviewing schedule, which was intractable here because of the high number of KC combinations in the q-matrix. Their results were then computed relatively to the optimal schedule. Thus, we are only looking for similar patterns, not comparing raw performance metrics.

We observe similar patterns of results between the different left-hand side figures, and these patterns are consistent with the results of Khajah et al. (2014): the ACP_R is very stable between $\theta = 0$ and $\theta = 0.5$ but decreases very rapidly afterward. This decrease is sharper for the multi-KC version of θ-*threshold*. These results may stem from the trade-off between proposing too difficult items (that the students are most likely to fail, resulting in lower further retention) and proposing too easy items (making the student only practice what they already know).

In the right-hand side figures, we see that high values (red curves) of θ tend to focus on already known KCs – especially KCs that were learned in the first weeks. However, this choice has diminishing returns. This could come from a "snowball" effect: at first, the strategies can only select KCs that were introduced in the beginning. For all θ parameters, these KCs are starting to be well-known, and their correctness probability is high. But as the student progresses, the strategies with the lowest θ get better than those with the highest θ. The highest θ stay stuck at making students review already mastered KCs and neglect the newly introduced KCs. For them, the forgetting of the oldest KCs may not be enough to select the most recent KCs.

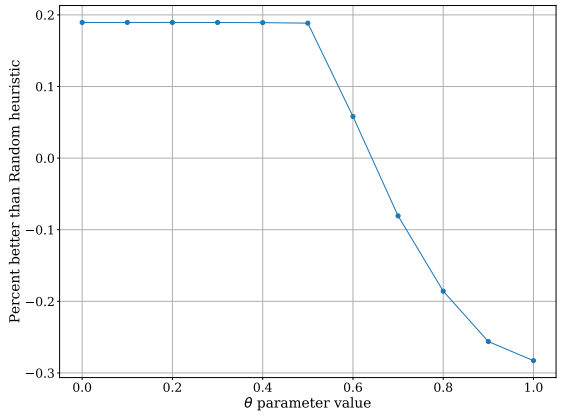


(a) Average *relative* ACP_R score comparison for different values of θ

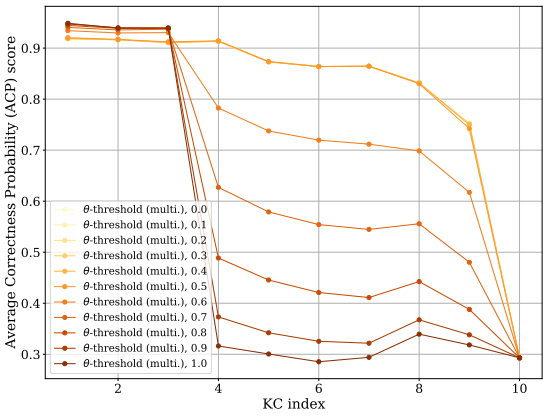


(b) ACP_R score comparison, broken down by KC index and θ parameter

Figure 7: Impact of the θ parameter value on the ACP_R score (*θ-threshold*)

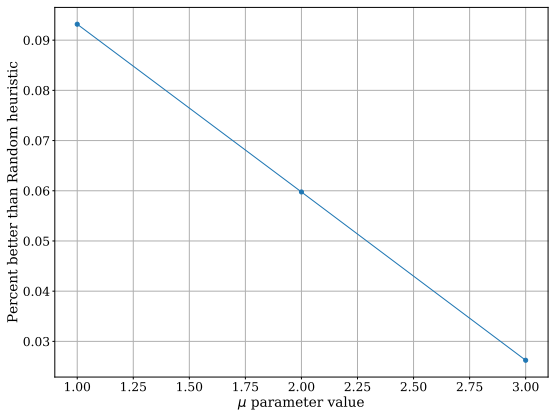


(a) Average *relative* ACP_R score comparison for different values of θ

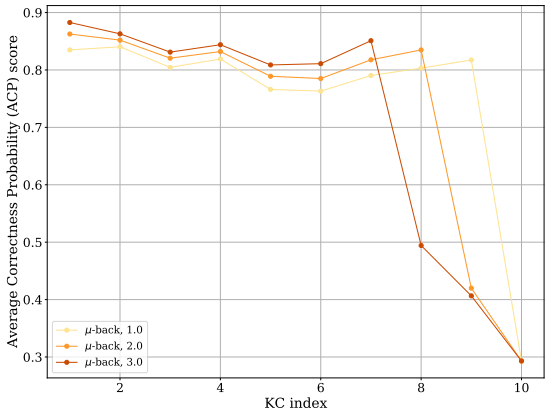


(b) ACP_R score comparison, broken down by KC index and θ parameter

Figure 8: Impact of the θ parameter value on the ACP_R score (*θ-threshold (multi.)*)



(a) Average *relative* ACP_R score comparison for different values of μ



(b) ACP_R score comparison, broken down by KC index and μ parameter

Figure 9: Impact of the μ parameter value on the ACP_R score (μ-*back*)

μ-BACK We performed the same type of analysis with the μ parameter (μ-*back*). Results are plotted in Figure 9.

In the left-hand side figure, we see that the performance decrease is linear in μ. In the right-hand side figure, we can see that the *3-back* strategy knows a sharp decrease after KC 7 because it is the last week that it was able to review; we observe the same decrease (with the same shape) but 1 (respectively 2) KCs after for the *2-back* and *1-back* strategies. Since *1-back* outperforms its counterparts, its ability to review KCs from the end of the curriculum must compensate for the loss from not reviewing enough the first few KCs.

In this Subsection, we compared the impact of a grid of hyperparameters for the θ-*threshold* and μ-*back* heuristics. We saw that θ-*threshold* strategies are stable for a wide range of θ values (between $\theta = 0$ and $\theta = 0.5$), whereas μ-*back* is much more sensitive to the choice of its μ parameter.

7.3. Real-World Implementation Challenges

In this article, we ran experiments on simulated students to compare the performance of different adaptive spacing algorithms for optimizing long-term mastery of a set of KCs. Even though we designed our experimental framework to make it as realistic as possible, we had to make a few simplifying assumptions. In this Subsection, we discuss two of these assumptions and the related challenges that a real-world implementation of our experiments and our algorithms would pose.

Independence of the KCs Throughout this study, we assumed that KCs were mutually independent. This may be seen as a strong assumption behind our experiments. This issue is rarely discussed in the literature when a KC-based model is used. The fact is that in a real situation, both the design of the set of KCs and the definition of the item-KC relationships constitute a difficult task. Especially, designing a set of atomistic KCs such that any student can practice each KC independently from the others is challenging. If there is leakage from one KC to another through exercising some items, algorithms will neither notice nor take this information into account.

If we have access to information regarding partial dependence between KCs, then it would

be possible to modify our algorithms to take this into account. One simple way to adapt to this issue could be to modify the underlying student model of *θ-threshold* and *Greedy* to account for dependency between KCs.

Identical Time Costs In our experiments, we assumed that each answer would take the same time for any learner, item, or KC. Instead, we allocated a fixed amount of practice items to all learners, like for a weekly worksheet in a classroom. In particular, a correct answer takes the same time as an incorrect answer in our experiments. This assumption of identical time costs is common in the adaptive spacing literature (Lindsey et al., 2014; Reddy et al., 2017; Upadhyay et al., 2018) but some research works have underlined its limits (Pavlik and Anderson, 2008; Eglington and Pavlik, 2020).

Indeed, in a real-world situation, items that are difficult to solve (e.g., because they are inherently difficult or involve KCs that have not been regularly practiced) may take more time to solve than easier items. In this case, if we wish to optimize mastery given a limited time budget (e.g., one weekly hour), the performance of some of our algorithms could change. For instance, let's consider *θ-threshold*. In Subsection 7.2, we saw that this strategy performed best for $\theta \in [0, 0.5]$, which means that the algorithm selects KCs that are difficult to remember. However, if we take time costs into account, items involving KCs selected by a lower θ would take more time to solve, and the optimal θ could be higher because it would lead to more practice opportunities within a given time budget (Eglington and Pavlik, 2020).

On the other hand, *Greedy* could easily be adapted to this different assumption. One way would be to modify its objective function by dividing it by the expected time cost of selecting each KC. This would require an additional model of response latency.

8. Conclusion and Further Work

8.1. General Conclusion

In this article, we investigated the research problem of optimizing long-term student mastery of KCs with the help of adaptive and personalized spacing algorithms. Until now, adaptive spacing algorithms have been designed for reviewing single items. We extended this standard framework for learning and reviewing a set of KCs. In our new setting, single items are used to practice one or multiple KCs at the same time. This property allows us to further accelerate student learning and memorization since we assume that practicing one single item improves memory retention of multiple KCs.

We first built an experimental protocol for comparing adaptive spacing heuristics. This protocol is realistic, and it relies on synthetic learning and forgetting trajectories. For this purpose, we used the DAS3H student model (Choffin et al., 2019). The DAS3H parameters, as well as the q-matrices of each of the simulation runs, were randomly generated according to specified probability distributions, in order to induce a large variability of learning and forgetting behaviors while controlling the general characteristics of these behaviors.

Based on the literature review that we conducted, we adapted to our research setting three spacing heuristics, originally designed for reviewing flashcards. We also proposed a new greedy procedure for iteratively selecting the most promising subset of KCs instead of the single best KC at any given timestamp. Thanks to this procedure, we extended two of our heuristics to

select multiple KCs at once. Before moving to a quantitative comparison of their performance, we qualitatively compared the characteristics of these five different heuristics.

Afterward, we conducted multiple experiments on 500 simulated students to compare the effectiveness of these KC selection heuristics on student long-term memory retention. To improve the robustness of our results, we ran these simulations 100 times, with each time a different randomly generated set of parameters. We implemented all the experiments in Python, from the simulation protocol to the spacing strategies. This code has been made public on GitHub. We showed with these experiments that a simple non-adaptive strategy such as *μ-back* is almost as effective as the adaptive spacing heuristics in most situations. Although similar to *μ-back* in terms of long-term KC mastery, *θ-threshold* has the advantage of improving the learner's correctness probability during the learning period, not just *after*. Conversely, our *Greedy* algorithm showed poor performance during learning and was only slightly better than all other single KC selection strategies during the retention period. However, it proved more robust to variability of the learning and forgetting behaviors than all other single KC selection strategies. Finally, we also showed that the greedy KC subset selection procedure allowed compatible heuristics (*Greedy* and *θ-threshold*) to outperform all single KC selection strategies.

8.2. Further Work

As a further work, we would like to perform the same type of strategy comparisons but with different student learning and forgetting models. Previous work has shown that equally performing student models could yield significantly different teaching sequences (Rollinson and Brunskill, 2015; Doroudi et al., 2017). More precisely, we wish to test the impact of model mismatch between the student simulator and the models used for scheduling reviews (*Greedy* and *θ-threshold* algorithms). We could follow the Robust Evaluation Matrix method from Doroudi et al. (2017) for this purpose. However, to the best of our knowledge, no current student model (apart from DAS3H) both allows inferring the impact of practicing an item on a set of underlying KCs and incorporates the forgetting effect within its structure. Thus, a first step in this direction would be to develop other student learning and forgetting models that would, like DAS3H (Choffin et al., 2019), allow us to infer KC mastery dynamics.

We could also account for other human cognitive properties in our models to further improve our simulations' realism. For instance, it seems that the benefits of the testing effect decrease with the complexity of a task to solve (Van Gog and Sweller, 2015). If we consider the number of KCs involved by an item to be a reliable proxy for the complexity of a task, then our simulation model DAS3H does not explicitly take this effect into account. Similarly, selecting only complex items could discourage learners. In that case, it would sometimes be preferable to choose items that are labeled by fewer KCs. In our work, we also made the simplifying assumption that all items took the same amount of time to solve. Instead of providing our algorithms with a fixed budget of practice items, we could allot them a fixed total *time* budget (Pavlik and Anderson, 2008) to choose the best KCs given these constraints.

We would also like to investigate the following research question: can we *learn* an optimal policy for optimizing KC long-term retention by interacting with synthetic students? In the past, several research works (Reddy et al., 2017; Yang et al., 2020; Upadhyay et al., 2018) have proposed deep reinforcement learning algorithms for optimally scheduling item reviews. It would be an interesting future work direction to develop similar algorithms for our research problem of reviewing KCs when items can involve multiple KCs at the same time, and to compare

these algorithms to the heuristics that we developed in this article.

Afterward, we would like to conduct a real-world controlled experiment to see if our simulated findings hold for helping real students practice and review real KCs. Our computational simulations allowed us to compare the proposed heuristics in an idealized setting, while controlling the learning and forgetting behaviors of our synthetic students. They also helped us to determine optimal parameter subsets for θ-*threshold* and μ-*back*, like Khajah et al. (2014) and Lindsey et al. (2014). Testing the adaptive spacing strategies that we developed on real human learners is however necessary to determine their impact on student performance and long-term mastery. Other adaptive spacing algorithms have been compared to non-adaptive schedulers by the past (Lindsey et al., 2014; Metzler-Baddeley and Baddeley, 2009) and have shown substantial improvement of memory retention at immediate and delayed tests.

9. Acknowledgments

This work was funded by Caisse des Depôts et Consignations, e-FRAN program.

References

Barzagar Nazari, K. and Ebersbach, M. 2019. Distributing mathematical practice of third and seventh graders: Applicability of the spacing effect in the classroom. *Applied Cognitive Psychology 33,* 2, 288–298.

Bjork, R. 1994. Memory and meta-memory considerations in the training of human beings. In *Metacognition: Knowing about knowing*, J. Metcalfe & A. Shimamura (Eds.), Ed. MIT Press, Cambridge, MA, US, 185–205.

Cepeda, N. J., Vul, E., Rohrer, D., Wixted, J. T., and Pashler, H. 2008. Spacing effects in learning: A temporal ridgeline of optimal retention. *Psychological Science 19,* 11, 1095–1102.

Choffin, B., Popineau, F., Bourda, Y., and Vie, J. 2019. DAS3H: modeling student learning and forgetting for optimally scheduling distributed practice of skills. In *Proceedings of the 12th International Conference on Educational Data Mining*, M. C. Desmarais, C. F. Lynch, A. Merceron, and R. Nkambou, Eds. International Educational Data Mining Society, 29–38.

Doroudi, S., Aleven, V., and Brunskill, E. 2017. Robust evaluation matrix: Towards a more principled offline exploration of instructional policies. In *Proceedings of the Fourth ACM Conference on Learning@Scale*. ACM, 3–12.

Doroudi, S., Aleven, V., and Brunskill, E. 2019. Where's the reward? *International Journal of Artificial Intelligence in Education 29,* 4, 568–620.

Dunlosky, J., Rawson, K. A., Marsh, E. J., Nathan, M. J., and Willingham, D. T. 2013. Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology. *Psychological Science in the Public Interest 14,* 1, 4–58.

Ebbinghaus, H. 1885. *Ueber das Gedächtnis*. Duncker and Humblot.

Effenberger, T., Pelánek, R., and Čechák, J. 2020. Exploration of the robustness and generalizability of the additive factors model. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge (LAK 2020)*. ACM, 472–479.

Eglington, L. G. and Pavlik, Jr, P. I. 2020. Optimizing practice scheduling requires quantitative tracking of individual item performance. *npj Science of Learning 5,* 1, 15.

Goutte, C., Durand, G., and Léger, S. 2018. On the learning curve attrition bias in additive factor modeling. In *Proceedings of the 19th International Conference on Artificial Intelligence in Education*. LNCS, vol. 10948. Springer, 109–113.

Hunziker, A., Chen, Y., Mac Aodha, O., Rodriguez, M. G., Krause, A., Perona, P., Yue, Y., and Singla, A. 2019. Teaching multiple concepts to a forgetful learner. In *Advances in Neural Information Processing Systems*. 4050–4060.

Kang, S. H., Lindsey, R. V., Mozer, M. C., and Pashler, H. 2014. Retrieval practice over the long term: Should spacing be expanding or equal-interval? *Psychonomic Bulletin & Review 21,* 6, 1544–1550.

Khajah, M. M., Lindsey, R. V., and Mozer, M. C. 2014. Maximizing students' retention via spaced review: Practical guidance from computational models of memory. *Topics in Cognitive Science 6,* 1, 157–169.

Leitner, S. 1972. *So Lernt Man Lernen*. Angewandte Lernpsychologie - ein Weg zum Erfolg. Herder.

Lindsey, R. V., Shroyer, J. D., Pashler, H., and Mozer, M. C. 2014. Improving students' long-term knowledge retention through personalized review. *Psychological Science 25,* 3, 639–647.

Mettler, E., Massey, C. M., and Kellman, P. J. 2016. A comparison of adaptive and fixed schedules of practice. *Journal of Experimental Psychology: General 145,* 7, 897–917.

Metzler-Baddeley, C. and Baddeley, R. J. 2009. Does adaptive training work? *Applied Cognitive Psychology 23,* 2, 254–266.

Pashler, H., Cepeda, N., Lindsey, R. V., Vul, E., and Mozer, M. C. 2009. Predicting the optimal spacing of study: A multiscale context model of memory. In *Advances in Neural Information Processing Systems 22*. Curran Associates, Inc., 1321–1329.

Pavlik, P., Bolster, T., Wu, S.-M., Koedinger, K., and Macwhinney, B. 2008. Using optimally selected drill practice to train basic facts. In *International Conference on Intelligent Tutoring Systems (ITS 2008)*. LNCS, vol. 5091. Springer, 593–602.

Pavlik, P. I. and Anderson, J. R. 2008. Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied 14,* 2, 101–117.

Pimsleur, P. 1967. A memory schedule. *The Modern Language Journal 51,* 2, 73–75.

Reddy, S., Labutov, I., Banerjee, S., and Joachims, T. 2016. Unbounded human learning: Optimal scheduling for spaced repetition. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, Eds. ACM, 1815–1824.

Reddy, S., Levine, S., and Dragan, A. 2017. Accelerating human learning with deep reinforcement learning. In *NIPS'17 Workshop: Teaching Machines, Robots, and Humans*.

Roediger III, H. L. and Karpicke, J. D. 2011. Intricacies of spaced retrieval: A resolution. In *Successful Remembering and Successful Forgetting*, A. Benjamin, Ed. Psychology Press, 41–66.

Rollinson, J. and Brunskill, E. 2015. From Predictive Models to Instructional Policies. In *Proceedings of the Eighth International Conference on Educational Data Mining*, O. C. Santos, J. Boticario, C. Romero, M. Pechenizkiy, A. Merceron, P. Mitros, J. M. Luna, M. C. Mihaescu, P. Moreno, A. Hershkovitz, S. Ventura, and M. C. Desmarais, Eds. International Educational Data Mining Society, 179–186.

Rowland, C. A. 2014. The effect of testing versus restudy on retention: a meta-analytic review of the testing effect. *Psychological Bulletin 140,* 6, 1432–1463.

Stamper, J., Niculescu-Mizil, A., Ritter, S., Gordon, G., and Koedinger, K. 2010. Algebra I 2005-2006 and Bridge to Algebra 2006-2007. Development data sets from KDD Cup 2010 Educational Data Mining Challenge. Find them at http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp.

Tabibian, B., Upadhyay, U., De, A., Zarezade, A., Schölkopf, B., and Gomez-Rodriguez, M. 2019. Enhancing human learning via spaced repetition optimization. *Proceedings of the National Academy of Sciences 116,* 10, 3988–3993.

Tatsuoka, K. K. 1983. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement 20,* 4, 345–354.

Upadhyay, U., De, A., and Rodriguez, M. G. 2018. Deep reinforcement learning of marked temporal point processes. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., 3168–3178.

Van Gog, T. and Sweller, J. 2015. Not new, but nearly forgotten: the testing effect decreases or even disappears as the complexity of learning materials increases. *Educational Psychology Review 27,* 2, 247–264.

Van Rijn, H., van Maanen, L., and van Woudenberg, M. 2009. Passing the test: Improving learning gains by balancing spacing and testing effects. In *Proceedings of the 9th International Conference of Cognitive Modeling*. Vol. 2. 7–6.

Vie, J. and Kashima, H. 2019. Knowledge tracing machines: Factorization machines for knowledge tracing. In *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. AAAI Press, 750–757.

Vlach, H. A. and Sandhofer, C. M. 2012. Distributing learning over time: the spacing effect in children's acquisition and generalization of science concepts. *Child Development 83,* 4, 1137–1144.

Wang, Z., Lamb, A., Saveliev, E., Cameron, P., Zaykov, Y., Hernández-Lobato, J. M., Turner, R. E., Baraniuk, R. G., Barton, C., Jones, S. P., Woodhead, S., and Zhang, C. 2020. Diagnostic questions: The NeurIPS 2020 education ehallenge.

Weinstein, Y., Madan, C. R., and Sumeracki, M. A. 2018. Teaching the science of learning. *Cognitive Research: Principles and Implications 3,* 1, 2.

Wozniak, P. and Gorzelanczyk, E. J. 1994. Optimization of repetition spacing in the practice of learning. *Acta Neurobiologiae Experimentalis 54*, 59–59.

Yang, Z., Shen, J., Liu, Y., Yang, Y., Zhang, W., and Yu, Y. 2020. TADS: learning time-aware scheduling policy with dyna-style planning for spaced repetition. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, J. Huang, Y. Chang, X. Cheng, J. Kamps, V. Murdock, J. Wen, and Y. Liu, Eds. ACM, 1917–1920.