Leveraging AI for Effective To-Do List Gamification

Saksham Consul
Max Planck Institute for Intelligent Systems, Tübingen
Germany
sakshamconsul@gmail.com

Jugoslav Stojcheski
Max Planck Institute for Intelligent Systems, Tübingen
Germany
jugoslav.stojcheski@gmail.com

Falk Lieder
Max Planck Institute for Intelligent Systems, Tübingen
Germany
falk.lieder@tuebingen.mpg.de

ABSTRACT

Many people procrastinate and struggle to prioritize their most important work. To help their users overcome such problems, gamified productivity tools like Habitica use heuristic point systems that can be counterproductive. We recently proposed a more principled way to compute point values that avoids such problems. Although it was promising in theory, it required large amounts of computation even for very short to-do lists. Here, we present a scalable approximate method that makes our principled approach to to-do list gamification useable in the real world. Our method leverages artificial intelligence to generate a gamified to-do lists, where each task is incentivized by a number of points that communicates how valuable it is in the long-run. What makes our new method more scalable is that it decomposes the problem of computing long-term plans for how the user can best achieve their goals into a hierarchy of smaller planning problems. We assessed the scalability of our method by applying it to to-do lists with increasingly larger numbers of goals, sub-goals, and tasks, and we also increased the number of nested levels of the goal hierarchy. We found that the method can enable web and mobile applications to compute excellent point systems for fairly large to-do lists, with up to **576** tasks spread out over up to 9 different top-level goals. Our method freely available through an API[1]. This makes it easy to use our method in gamified web applications and mobile apps.

KEYWORDS

gamification, productivity tools, prioritization, AI, procrastination

1 INTRODUCTION

Procrastination and prioritization are challenges that many people face in their daily lives [20] because setting the right priorities and working diligently requires a lot of mental effort and self-discipline. These challenges decrease people's productivity, which is defined as the amount of value that a person generates by completing a series of tasks within a fixed amount of time.

According to temporal motivation theory [21], people procrastinate because distant outcomes are less motivating than immediate outcomes [20]. As a consequence, the motivational pull of highly valued distant outcomes (e.g., completing one's dissertation) is often lower than the desire to avoid unpleasant emotions and the appeal of small immediate pleasures (e.g., the enjoyment of a funny video). This is problematic when immediate experience of working on a project is misaligned with its long-term value. Some productivity apps, such as Habitica, attempt to rectify this problem using points, badges, and levels. These game elements can be used to promote specific behaviors by increasing people's extrinsic motivation [14, 15]. However, awarding points, levels, and badges in such a way that people become more productive is tricky [5, 13]. Adding such game elements does not foster intrinsic motivation per se [14, 15]. Moreover, previous research found that even intuitively reasonable point systems can inadvertently incentivize counterproductive behaviors or undermine the users' motivation [5, 13, 17, 23]. To address this challenge, we previously applied temporal motivation theory [21] to develop a principled mathematical framework for designing point systems that incentivize each task in proportion to how valuable it is for the user in the long-run [10]. Controlled online experiments suggested that this approach can help people overcome procrastination and become more productive [10, 11]. So far, this approach has been limited to artificial scenarios with short to-do lists because the time it takes to compute the optimal number of points increases exponentially fast with the number of tasks in the to-do list. Therefore, the first goal of this paper is to mitigate this problem so that to-do list gamification can be applied to real-world to-do lists.

Given that people typically have to juggle many goals at any given point of time [12], it comes as no surprise that they often struggle to identify and prioritize what is most important for their goals and projects in the long run. Instead, they often get distracted or side-tracked by less important tasks that are more urgent in the short-run. This can cause important long-term projects to get derailed and individuals and their managers and organizations to become frustrated. While existing productivity tools help people remember their tasks and goals, they rarely help people prioritize what is most important in the long run. The few systems that do, put most of the burden of figuring out what should be prioritized on the user or are only applicable in specific domains [1, 3, 6, 18, 19]. Therefore, the second goal of this paper is to extend the optimal to-do list gamification approach introduced by [10] so that it can be used to help people decide which goals to prioritize.

To achieve these goals, we leverage efficient hierarchical planning algorithms that were developed in artificial intelligence to make the optimal to-do list gamification method proposed by [10] scalable. Our approximate method exploits that people's goals, subgoals, and tasks are organized hierarchically (see Figure 1) by using a 2-level hierarchical decomposition of a discrete time semi-Markov decision process (SMDP; [7]). Our method supports goal systems

[1]Link to Code

Veröffentlicht durch die Gesellschaft für Informatik e.V.
in K. Marky, U. Grünefeld & T. Kosch (Hrsg.):
Mensch und Computer 2022 – Workshopband, 04.-07. September 2022, Darmstadt
© 2022 Copyright held by the owner/author(s).
https://doi.org/10.18420/muc2022-mci-ws08-241

with many hierarchically nested levels of subgoals. It allows the user to indicate that some sub-goals or tasks are essential to the completion of the corresponding superordinate goals, and to assign different levels of importance to different goals and tasks. Our results show that the resulting method is much faster than the exact method, but still very accurate. It can therefore be applied in the real world.

The plan for this paper is as follows. We first illustrate the idea of optimal to-do list gamification with an example. We then formalize the goal of optimal to-do list gamification in the language of artificial intelligence. We then present the computational method we developed to solve this problem. Next, we evaluate our method's accuracy and scalability on a series of benchmark problems. We close by discussing the implications of our findings and directions for future research.

2 AN ILLUSTRATIVE EXAMPLE OF OPTIMAL TO-DO LIST GAMIFICATION

To explain the idea of optimal to-do list gamification, we now describe a realistic use case and present the gamified to-do list our method generates in this example.

The user's inputs: Imagine the user has three goals: Goal A, Goal B, and Goal C. Goal A is the most urgent one because it is due in 12 time steps and takes 12 time steps to accomplish. It can only be completed on time if all its tasks are prioritized first. Goal B has a more flexible deadline and can be completed even after Goal A has been completed. Goal C is a goal which has a high value, but it cannot be completed by its deadline. Each of those goals has two subgoals, and each subgoal comprises multiple tasks.

In the hierarchical to-do list below, the `Value` of a goal represents how valuable it is for the user to achieve it. Its `Deadline` indicates how much time the user has to achieve the goal. Each sub-goal has an `Int Reward` which denotes how valuable completing the sub-goal is to the user, independent of the corresponding goal. `Ess` is a Boolean value which is `True` if the sub-goal is deemed essential to complete the corresponding goal. Additionally, `Imp` denotes how important the sub-goal is relative to the other sub-goals. Finally, `Time` indicates the estimated time to complete a sub-goal.

```
Goal A, - Value: 1000, Deadline: 12,

    SG A1 - Int Reward: 40, Ess: True,Imp: 100, Time: 7
      Task A11 - Int Reward: 10, Ess: True,Imp: 60, Time: 3
      Task A12 - Int Reward: 15, Ess: True,Imp: 20, Time: 2
      Task A13 - Int Reward: 15, Ess: True,Imp: 20, Time: 2

    SG A2 - Int Reward: 30, Ess: True,Imp: 100, Time: 5
      Task A21 - Int Reward: 20, Ess: True,Imp: 60, Time: 3
      Task A22 - Int Reward: 2, Ess: True,Imp: 30, Time: 1
      Task A23 - Int Reward: 8, Ess: True,Imp: 10, Time: 1

Goal B, - Value: 500, Deadline: 50,

    SG B1 - Int Reward: 100, Ess: True,Imp: 100, Time: 6
      Task B11 - Int Reward: 80, Ess: True,Imp: 90, Time: 4
      Task B12 - Int Reward: 20, Ess: True,Imp: 10, Time: 2

    SG B2 - Int Reward: 100, Ess: True,Imp: 100, Time: 17
      Task B21 - Int Reward: 20, Ess: True,Imp: 20, Time: 2
      Task B22 - Int Reward: 10, Ess: True,Imp: 60, Time: 2
      Task B23 - Int Reward: 10, Ess: True,Imp: 2, Time: 1
      Task B24 - Int Reward: 40, Ess: True,Imp: 15, Time: 10
      Task B25 - Int Reward: 20, Ess: True,Imp: 3, Time: 2

Goal C, - Value: 5000, Deadline: 50,

    SG C1 - Int Reward: 10, Ess: True,Imp: 100, Time: 3
      Task C11 - Int Reward: 5, Ess: True,Imp: 60, Time: 1
      Task C12 - Int Reward: 5, Ess: True,Imp: 40, Time: 2

    SG C2 - Int Reward: 90, Ess: True,Imp: 100, Time: 502
      Task C21 - Int Reward: 50, Ess: True,Imp: 20, Time: 50
      Task C22 - Int Reward: 10, Ess: True,Imp: 60, Time: 400
      Task C23 - Int Reward: 20, Ess: True,Imp: 10, Time: 50
      Task C24 - Int Reward: 10, Ess: True,Imp: 10, Time: 2
```

The output of our optimal to-do list gamification method: Our method assign points in such a manner that a person who always chooses the task with the largest number of points will always do what is best for them in the long run. In this example, the optimal sequence of tasks in the to-do list is to first select all the tasks to achieve Goal A because it has the highest value (1000) and a tight deadline. Since all tasks of Goal A are essential, all tasks of Goal A are first selected. Because completing the first subgoal of Goal A takes the user closer to achieving Goal A than completing its second subgoal (7 vs. 5 time steps) and has higher intrinsic value to the user (Int Reward 40 vs. 30), its tasks (Task A11-A13) are assigned more points than the tasks of the second subgoal (Task A21-A23). Task A11 was assigned slightly fewer points than Tasks A12 and A13 because the user indicated that it has less intrinsic value to them (Int Reward 10 vs. 15).

Next, the tasks to achieve Goal B were selected. After that, only the tasks needed to accomplish Goal C remain. At this stage, the optimal action to perform would be to slack-off. This makes sense and can be explained by looking into the sub-goals of Goal C. Goal C consists of 2 essential sub-goals, of which sub-goal SG C2, cannot be completed before the deadline, hence, performing any tasks of Goal C would not yield to the completion of Goal C and should not be completed.

Therefore, before the user has completed any of their tasks, the output of our method looks like this:

```
Task A12: 1574.5 Points
Task A13: 1574.5 Points
Task A11: 1574.3 Points
Task A23:  322.8 Points
Task A22:  322.8 Points
Task A21:  322.8 Points
Task B25:  282.8 Points
Task B21:  282.8 Points
Task B23:  282.8 Points
Task B22:  282.8 Points
Task B24:  282.6 Points
Task B11:   94.0 Points
Task B12:   94.0 Points
Slack off:  10.1 Points
Task C11:    7.0 Points
Task C12:    7.0 Points
Task C24: -399.8 Points
Task C21: -399.9 Points
Task C23: -400.0 Points
Task C22: -402.9 Points
```

After the user has completed the tasks of Goal A and Goal B on time, the gamified to-do list generated by our method would look as follows:

```
Slack off:  10.1 Points
Task C11:    7.0 Points
Task C12:    7.0 Points
Task C24: -399.8 Points
Task C21: -399.9 Points
Task C23: -400.0 Points
Task C22: -402.9 Points
```

As you can see, the gamified to-do list generated by our method now encourages the user to slack off instead of working on the tasks for Goal C.

3 PROBLEM FORMULATION

The goal of to-do list gamification is to maximize the user's long-term productivity by proposing daily to-do lists where each task is incentivized by a certain number of points. According to temporal motivation theory [21], we can help people overcome procrastination by distributing the distant value of the goal over more proximal and more attainable milestones. Therefore, the goal of optimal gamification is to assign points to tasks and sub-goals in such a way that people are motivated to complete the tasks that are most valuable to them in the long run. Regardless of whether the points instill an expectation of extrinsic or intrinsic value and what the underlying psychological mechanisms might be, we assume that the points people anticipate for their immediate next task are more motivating to them than the prospect of receiving points for other tasks later.

Users compose hierarchical to-do lists comprised of three types of items: *goals*, *sub-goals* and *tasks*. We define a *task* as a sub-goal that cannot be further decomposed. Simply put, a task is the smallest non-divisible unit of work.

As described in Section 3.1, we first model the sequences of tasks that user's pursuit of their goals as a discrete-time semi-Markov Decision Process (SMDP) [7]. In brief, a semi-Markov Decision Process is a mathematical model of the problem of choosing a sequence of actions (here tasks) that can take different amounts of time in such a way as to maximize the sum of the subjective utility attained by completing tasks and achieving (sub)goals ("reward") minus the sum of the action's costs. Please note that we are using the word "reward" to refer to a content-free technical construct of the SMDP formalism. We use the reward function of the SMDP as a placeholder for any positive or negative subjective value of any external or internal event of any kind. This is not limited to extrinsic incentives such as money, but also includes the intrinsic enjoyment that people derive from certain activities. Therefore, our formal mathematical does not constitute a commitment to behaviorism. To the contrary, it is fully compatible with cognitive theories of (intrinsic) motivation [see 16].

Solving the SMDP model of a user's projects generates the optimal plan to complete a list of user-specified tasks. Our method approximates the solution of the large SMDPs by treating each level of the hierarchical to-do list as a mini-SMDP. Each mini-SMDP consists of 1 *goal* and multiple *sub-goals*. In each mini-SMDP, the sub-goal is treated as a task, and as such, will be referred to as a task from now on. We define a root of a to-do list as an imaginary goal, whose sub-goals are the user's top-level goals (see Figure 1). **Goals** consist of a deadline, a value estimate, and a list of sub-goals. Each **sub-goal** contains an estimate of how long it will take to complete it, a Boolean value to indicate if the sub-goal is essential to complete its corresponding goal, its importance for achieving the corresponding goal, and the intrinsic value of achieving the subgoal above and beyond its instrumental value for achieving the larger goal. A sub-goal is deemed as essential if the completion of the sub-goal is necessary for the completion of the goal. In other words, a goal cannot be completed without completing *all* of its essential sub-goals. As such, an essential sub-goal is marked with a high importance factor, and a non-essential sub-goal should be marked with a low importance factor. The intrinsic value of a task or subgoal is an optional parameter the user can specify to communicate to which extent they value completing a task or subgoal for its own sake. For instance, a student with the goal to get an A could use this parameter to express how much they value learning about a certain topic for its own sake, while leaving it blank for topics that they only care about to the extent that studying them improves their grade.

In addition, users specify their desired workload in hours for a typical day (*typical day's working hours*) and for the day at hand (*today's working hours*). The outputted gamified daily schedule should contain all tasks that users indicated they wanted to work on today, as well as additional tasks towards their goals, up to the desired daily workload.

3.1 Modelling the to-do-list as a discrete-time SMDP

Formally, an SMDP is defined by a set of states, a set of actions, a transition time function, a transition dynamics function, and a reward function. We will now define each of these components of our model in turn.

The state space $\mathcal{S}$ consists of all possible combinations of completed and uncompleted tasks. Since a task can either be completed or uncompleted, each $s \in \mathcal{S}$ is a binary vector of length $n+1$, where n corresponds to the total number of tasks in the SMDP and the slack-off task. The slack-off task represents when an user decides to no longer do any more productive work. If the i-th element of any binary vector $s \in \mathcal{S}$ is 1, the task associated with the element is completed, and 0 otherwise. Similarly, the action space $\mathcal{A}$ refers to the selection of a task and hence $a \in \mathcal{A}$ is of the same size, $n+1$.

$$\mathcal{S} = \{\{0,1\}^{n+1}\} \quad (1)$$

Considering G top-most goals, each goal having D sub-goal levels and each sub-goal having B sub-sub goals, the to-do list would have a total of $G \cdot B^D$ tasks. Without breaking the to-do list into

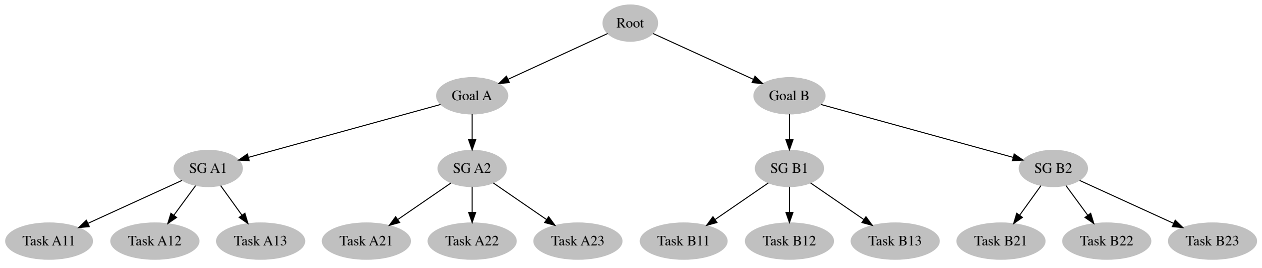

Figure 1: Graphical example of a hierarchical to-do-list.

multiple mini-SMDPs, the state space would be of size $2^{(G \cdot B^D+1)}$. By breaking down into mini-SMDPs, we would need to solve $G \cdot D$ SMDPs with a state space 2^{B+1}. Hence, the decomposition of the to-do list into mini-SMDPs shrink the size of the state space and the number of computations go down from $\mathbf{2^{(G \cdot B^D+1)}}$ to $\mathbf{B \cdot D \cdot 2^{B+1}}$.

3.1.1 Transition time F. In a SMDP setting, the transition-time function, $F(\tau|s_t, a)$ is the probability that the time at which the agent has to make the next decision occurs in exactly τ time units, as a consequence of executing action a in state s at the time t. We chose a transition-time function that can model the cognitive bias known as the *planning fallacy*, in which people underestimate the time required to complete a task. Kahneman and Tversky [8] describe this bias as such: "*Scientists and writers, for example, are notoriously prone to underestimate the time required to complete a project, even when they have considerable experience of past failures to live up to planned schedules... It frequently occurs even when underestimation of duration or cost is actually penalized.*"

Since the SMDPs are modelled with discrete time steps and people have unreliable time estimates, we model the number of time units required for action completion to follow a zero-truncated Poisson probability distribution[2] with adjusted mean value and variance. We formally define the transition-time function as

$$F(\tau|s_t, a) := \text{Poisson}_{>0}(\tau; \tilde{k}) = \frac{\tilde{k}^{\tau} e^{-\tilde{k}}}{\tau!(1 - e^{-\tilde{k}})}$$

where $\tilde{k} = c_{\text{pf}} \cdot k$, k is the discrete amount of time units required to complete action a in state s at the time t, and $c_{\text{pf}} \in \mathbb{R}_{>0}$ is a planning-fallacy constant that adjusts the distribution parameter. In lack of knowledge about the exact value of the planning-fallacy constant (c_{pf}), we follow King and Wilson [9] and we initially set its value to 1.39. Obtaining better estimates for this value based on real-world data is left for future work.

3.1.2 Transition dynamics T. The transition dynamics from a current state s at the time t to a next state s' after executing an action a is deterministic in completion, but stochastic in duration. In other words, the presented algorithm assumes that users will complete a task once they start, but may require more time than the time estimated for completion of the task. Formally, if an action a is represented by the i-th bit of the binary state vector s, the binary vector of the next state s' can be written as $s' = e_i \vee s$, where e_i is a one-hot vector with a value of 1 only at its i-th position, and $\vee$ represents the "or" operation of two binary vectors.

A special case of the transition dynamics occurs after reaching the terminal state in which all real tasks have been completed or if the slack-off action is selected. There, the process transits to a goal-achieving state $s_\dagger$ after instantaneously executing the action $a_\dagger$ in 0 time steps, that is $T(\mathbf{1}, a_\dagger, s_\dagger) = \Pr(s_\dagger|\mathbf{1}, a_\dagger) = 1$.

3.1.3 Reward function. We define the reward function $r(s_t, a, s'_{t+\tau})$ from a current state s at time t to the next state s' at time $t + \tau$ after performing action $a \in \mathcal{A}$ that takes τ time units for execution to be

$$r(s_t, a, s'_{t+\tau}) = \begin{cases} R(a_+) \cdot (1 - \gamma)^{-1}; \text{if the slack off action was chosen} \\ -\lambda^{(g)} \sum_{k=0}^{\tau-1} \gamma^k \\ +\gamma^{\tau-1} \cdot r_{\text{extrinsic}}(s_t, a, s'_{t+\tau}) \cdot \Pi(\beta_g) \\ +r_{\text{intrinsic}}(a); \text{if any other action was chosen} \end{cases}, \tag{2}$$

where $r_{\text{intrinsic}}$ represents how much the user intrinsically values completing the task for its own sake.

The extrinsic component ($r^{(g)}_{(\text{extrinsic})}$) of this reward function is

$$r^{(g)}_{(\text{extrinsic})}(s_t, a, s'_{t+\tau}) = \begin{cases} R(g) \cdot \frac{\sum I_{done}(s'_{t+\tau})}{\sum_{a_k \in \mathbb{A}^g} I(a_k)}; \text{if goal is complete} \\ 0; \text{if goal hasn't been completed} \end{cases}, \tag{3}$$

where a_+, represents *slack-off* action, $\gamma \in (0, 1]$ is a discount factor, $R(g)$ indicates the value of a goal $g \in \mathcal{G}$. $\lambda^{(g)} \in \mathbb{R}_{>0}$ models the value that reflects the cost of a person's time and mental effort to work on goal g. $I_{done}(s'_{t+\tau})$ refers to the list of importance values of the subset of completed tasks in state $s'_{t+\tau}$.

$R(g)$ returns the goal value if executing the next task-level action a leads to completion of goal g. We define $\Pi(\beta_g)$ to be the penalty function for a goal g. The value of the penalty function discounts the goal reward proportionally to the time by which deadlines associated with that goal are missed, and it can be formally written as $\Pi(\beta_g) = (1 + \beta_g)^{-1}$. Here $\beta_g = \sum_i^n \psi \cdot \Delta t_i$ is a weighted sum of penalties for tasks whose deadlines were $\psi \in \mathbb{R}_{>0}$ is the penalty rate (per unit time) and Δt_i is the number of time units by which the deadline was missed.

[2]Also known as *conditional Poisson distribution, positive Poisson distribution.*

According to the definition, an immediate negative reward and a small positive reward is obtained for completing each task. Conversely, an immediate positive reward is obtained *after* the goal is completed or a slack-off action has been chosen for execution. A goal is said to be completed if all *essential* sub-goals are completed.

3.1.4 *Optimal policy.* As described in Section 4.2, each to-do list is broken down into mini-SMDPs in a hierarchical manner, treating each layer as a goal to propagate and compute the value of each individual task. Once all the 2-layer mini-SMDPs have been solved, their tasks and the corresponding point values are collated into a single gamified to-do list. The (approximately) optimal policy can then be defined as always choosing the task with the largest number of points from the list of uncompleted tasks.

3.2 Maximizing the productivity of myopic workers by optimal gamification

The SMDP defined above allows us to formalize a person's productivity from time t_1 to time t_2 by

$$\mathrm{P}(s_{t_1}, s_{t_2}) = \frac{V^\star(s_{t_2}) - V^\star(s_{t_1})}{t_2 - t_1}, \tag{4}$$

where s_{t_1} and s_{t_2} describes the sets of tasks that the person had completed by time t_1 and time t_2 respectively.

Following [10], we model people as *myopic bounded agents* who generally choose tasks based on the difference between their immediate reward minus the subjective cost of working on a task, which we model as the product of the task's unpleasantness and its duration. That is, we assume that people select tasks according to the greedy policy

$$\pi_{\text{greedy}}(s) = \arg\max_a \mathbb{E}\left[r(s, a, s')\right]. \tag{5}$$

Under this assumption, the problem of maximizing people's productivity by optimal gamification can be formalized as computing optimal incentives $f^\star(s, a)$ so that

$$\forall s : \pi^\star(s) \in \arg\max_a \left\{f^\star(s, a) + \mathbb{E}\left[r(s, a, s')\right]\right\}. \tag{6}$$

Lieder et al. [10] proved that this can be achieved by setting $f(s, a)$ to the optimal incentives

$$f^\star(s, a) = \mathbb{E}\left[V^\star(s')|s, a\right] - V^\star(s) \tag{7}$$

$$= \max_{a'} Q^\star(s', a') - \max_{\tilde{a}} Q^\star(s, \tilde{a}) \tag{8}$$

To help people choose the most valuable task, their daily to-do list should include the tasks $\pi^\star(s_t), \pi^\star(s_{t+1}), \cdots$ that the optimal policy would choose on that particular day (where t is the first time step of the person's work day). Unfortunately, naive computation of $f^\star$ is intractable for real-world applications. The goal of this text is to present a scalable algorithm for approximate computation of optimal incentives $f^\star$ for real-world applications of productivity apps.

4 METHOD

An SMDP consists of one goal and multiple sub-goals. As mentioned in Section 3, while solving the SMDP, we consider each sub-goal as a task with no sub-tasks.

4.1 Solving mini-SMDP

Given a mini-SMDP with B sub-tasks and starting time as $t = t_o$, the stating state s_{t_o} is represented as a vector of size $B+1$. The mini-SMDP is computed by first checking the tasks completed, marking the vector 1 if the corresponding task is completed. The last index of the vector represents if the slack-off action has been previously selected, indicating that the state is now in state $s_\dagger$.

From the starting state, the method computes the expected reward for all the possible sequences of tasks. After initializing the Q-values for a given state and time, it iterates over all the possible actions in the given state. If the optimal sequence following an action is not computed before and if the action is not a slack-off action, the method computes the expected reward for following the given action and then the optimal policy before updating the Q-value for a given state and the expected reward of doing a task.

The expected reward for completing the task is computed by iterating over time estimates using the transition function F. The expected task reward is calculated by taking the weighted average of the reward obtained by completing a task for each time estimate. It takes into account the penalty of missing the deadline (if missed) and the discounted cumulative cost for performing the task an the immediate reward from the reward function $r(s_t, a, s'_{t'})$. Additionally, the reward for following the optimal policy is computed for each time estimate to also compute the expected total reward used Used for updating the Q-value.

4.2 Passing of the value function

To understand how the value is propagated down the SMDP, we utilize an example to-do list shown in Figure 1. It consists of 2 goals, which having 2 sub-goals. Each sub-goal has 3 tasks. The SMDP is broken down into 7 mini-SMDPs as shown below:

(1) Goal: Root, Sub-goals: Goal A, Goal B
(2) Goal: Goal A, Sub-goals: SG A1, SG A1
(3) Goal: Goal B, Sub-goals: SG B1, SG B2
(4) Goal: SG A1, Sub-goals: Task A11, A12, A13
(5) Goal: SG A2, Sub-goals: Task A21, A22, A23
(6) Goal: SG B1, Sub-goals: Task B11, B12, B13
(7) Goal: SG B2, Sub-goals: Task B21, B22, B23

A dummy "Root" node is created to facilitate solving of the mini-SMDP. The root goal value is the sum of values of the goals assigned by the user and denotes the total number of productivity value in the to-do list. The importance of each goal is given as the ratio between the value assigned to the goal and the sum of values of all goals.

In the 1st mini-SMDP, all goals are represented as tasks and marked as non-essential. The intrinsic value of a goal is computed recursively by computing the sum of the intrinsic values of all its sub-goals.

After solving the 1st mini-SMDP, the value of the sub-goals (Goal A, Goal B), denoted by $R(SG_k)$ will be passed down to the 2nd and 3rd mini-SMDPs as follows

$$R(SG_k) = \frac{e^{\eta(SG_k)}}{\sum_{i=1}^{n} e^{\eta(SG_i)}} \cdot \left[R(g) + \sum_{\forall SG_i \in P(g)} r_{\text{intrinsic}}(SG_i)\right] \tag{9}$$

with

$$\eta(SG_k) = \{\gamma^{\tau_{SG_k}} \cdot \mathbb{E}[V^*(s'|s, SG_k)] - V^*(s)\}$$
$$+r_{\text{extrinsic}}(s_t, SG_k, s'_{t+\tau_{SG_k}}) \cdot \Pi(\beta_{SG_k}) + r_{\text{intrinsic}}(s_t, SG_k, s'_{t+\tau_{SG_k}}) \tag{10}$$

where $R(g)$ is the return of the optimal policy of the 1^{st} mini-SMDP. τ_{SG_k} is the time estimate to complete all essential tasks of the sub-goal SG_k.

Similarly, after solving the 2nd mini-SMDP, the value of SG A1 will be passed down to the 4th mini-SMDP and value of SG A2 to 5th mini-SMDP. Likewise, the value of SG B1 and SG B2 will be passed down to the 6th and 7th mini-SMDP, respectively. The flow of solving mini-SMDPs is depicted in Figure 2

5 RESULTS

As detailed in the following two subsections, our method is both highly accurate and highly scalable. For an illustrative example of the gamified to-do lists that our method produces, see Section 2. The code for our API is available on Github.

5.1 Accuracy

To evaluate the quality of the point values computed by our method, we simulate their effect on a user who always selects a task with the highest number of points in the incentivized to-do list until the points for the task with the highest points is less than the slack-off reward. This strategy is called the myopic greedy strategy. We compare the performance of the myopic greedy strategy on to-do lists incentivized by the points computed by our proposed algorithm with its performance on the same to-do lists when they are incentivized by the exact point values calculated using Value Iteration [2]. The performance metric selected is the actual aggregated reward. The actual reward consists of the reward for completing a task (intrinsic value) and the reward for completing a goal (value associated with the goal). In case the deadline of a task or goal is missed, the value associated with the completion of the task or goal is not included.

We define a loss ratio (lr) metric to compare the performance of the proposed algorithm with the exact solution as follows:

$$\text{lr} = 100 \cdot \frac{(R_{\text{exact}} - R_{\text{algorithm}})}{R_{\text{exact}}}, \tag{11}$$

where R_{exact} and $R_{\text{algorithm}}$ are the returns that our model of a myopic worker achieves when the to-do list is incentivized by the points computed with the exact method and our new approximate method, respectively.

We assessed the accuracy of our method's approximate solution in 28 hand-crafted case studies. These case studies were designed to cover realistic use-cases with a typical number of goals and typical numbers of subgoals per goal and tasks per subgoal (e.g., 3 goals, 2 subgoals per goal, and 3 tasks per subgoal, 18 tasks in total). Additional test-cases systematically varied the number of goals and subgoals. We also included test cases where the number of tasks and subgoals differed across goals and test cases that systematically varied the values of the additional parameters that were not available in the earlier version by [22] (e.g., intrinsic values, importance, and whether a task is necessary to achieve the goal). The list of test cases can be found in Appendix A1 of [4]. A loss-ratio of 0 indicates that the performance of the sequence of tasks followed by using the myopic greedy strategy of the optimal gamified points computed by our proposed algorithm and the exact solution is the same. All 28 case studies yielded a loss-ratio of 0 which indicates that the calculation of the gamified points using our proposal yields the exact same performance as using the optimal gamified points.

5.2 Scalability

While there is no theoretical limit for the size of the to-do list for which the proposed algorithm can solve, there is a practical limit set by the services used to run the API. The API is hosted on a Heroku server, which has a practical 30 seconds time limit for the API request to be active. We compared the time required for solving a to-do list in its initial state with varying number of goals, maximum depth and the branching factor of the to-do list. The maximum depth of the to-do list is defined as the lowest level a task can be abstracted. The branching factor is the number of sub-goals a goal can be divided into. For example, the maximum depth of the to-do list illustrated in Figure 1 is 3 with a branching factor of 3. The tasks generated in the scalability assessment are all essential, require an estimated time of 1 time unit to be completed, have an intrinsic value of 1 and have equal importance to other tasks.

While Figure 3 shows that the time required scales linearly to the increase in the number of goals, Figure 4 and Figure 5 show that the time required grows exponentially to the increase in branching factor and maximum depth.

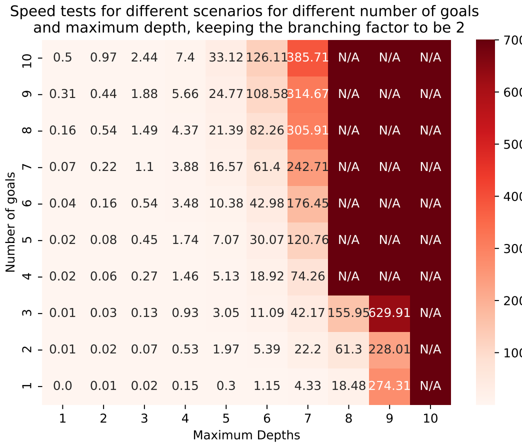

Figure 4: Heat-map representing the time taken (seconds) for case studies with varying number of goals and maximum depths with a branching factor of 2

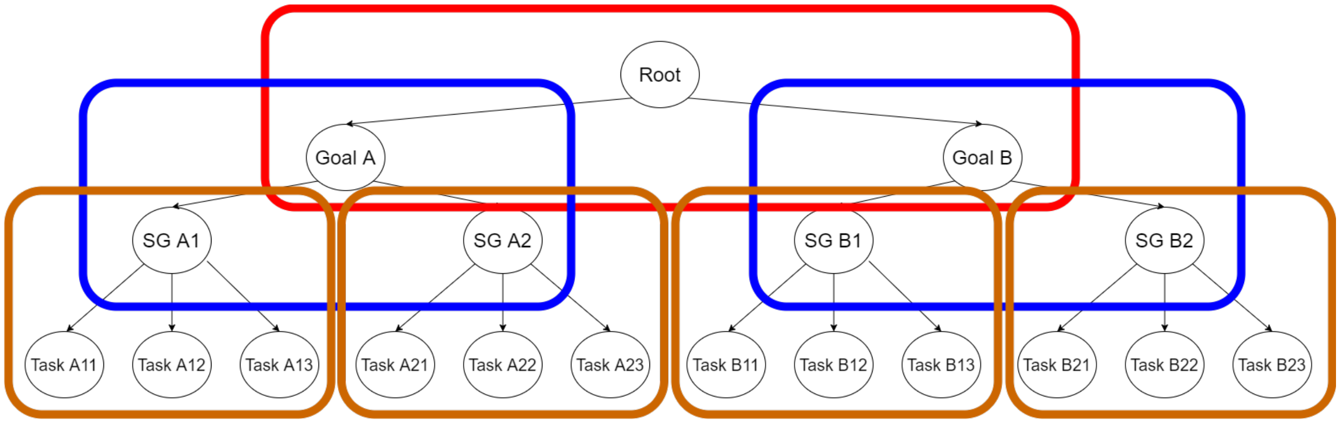

Figure 2: Graphical depiction of how the mini-SMDPs are solved. The mini-SMDP in red is solved first, followed by the ones in blue and finally the mini-SMDPs in brown are solved last.

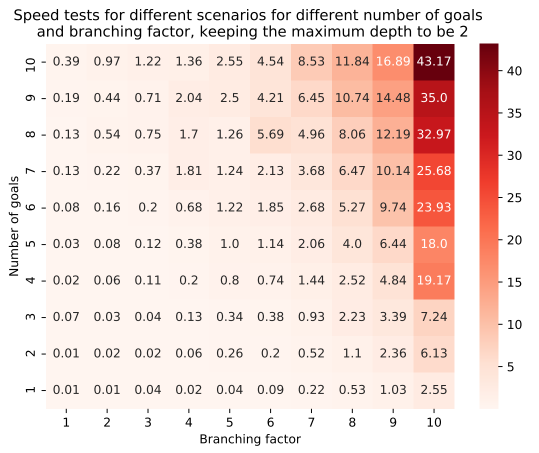

Figure 3: Heat-map representing the time taken (seconds) for case studies with varying number of goals and branching factors with a maximum depth of 2

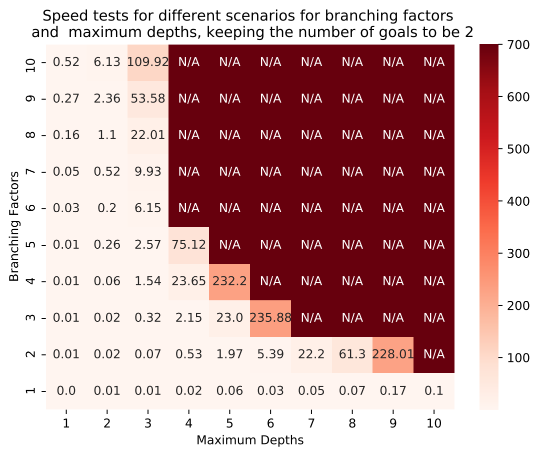

Figure 5: Heat-map representing the time taken (seconds) for case studies with varying number of branching factors and maximum depths and 2 goals

Even with such constraints, Figure 6 shows that a to-do list with a depth of 3 and branching factor of 4 is easily solvable by our proposed algorithm. Such a to-do list has a total of **576** tasks, which is more than big enough for most real-life examples. This shows that our proposed algorithm is practically useful.

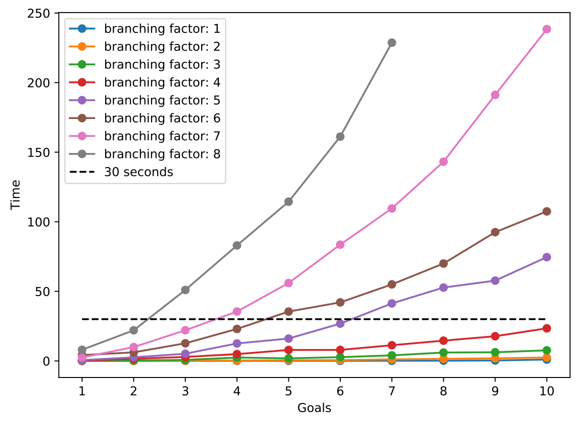

Figure 6: Speed-test to see the trend of increasing goals for a given branching factor with a maximum depth of 3

6 DISCUSSION

Summary and Interpretation. To improve productivity in complex situations with multiple goals, people need to plan rigorously and regularly re-evaluate their plan and potentially adjust it as the situation changes. This is very effortful, time-consuming, and prone to fall prey to human biases, such as the planning fallacy and the sunk-cost fallacy. Hence, a scientific and computationally efficient solution is required.

We devised a scalable AI-powered approach for assigning points to the tasks on a person's to-do list in such a way that the task that are most valuable for their long-term goals are also most appealing in the short-run. The central idea of our method is to decompose the problem hierarchically and computing the optimal plan at each level of the goal hierarchy.

Limitations and Future Directions. Since our approach decomposes the full sequential decision problem hierarchically, its accuracy can be limited by the fact that it never considers the whole problem at once. This is unproblematic for most general use cases, but might be suboptimal in certain situations. Additionally, in this article, we have only provided an analytical study of our algorithm's usefulness. Future studies should test how the points computed by our method affect the productivity, procrastination, and motivation of real people in the real world. Concretely, we are planning to conduct a longitudinal field experiment in which the experimental group is supported by a to-do list app that uses our optimal gamification method whereas the control groups either receive no points, randomly generated points, or points computed by a simple heuristic. Moreover, since the assumed value of slacking off affects the recommendations of our method, future research should try to measure this parameter empirically.

We are considering multiple potential ways to improve the functionality of our system to make it more useful in the real world. This includes supporting tasks that contribute to multiple goals simultaneously. Moreover, we are planning to further decrease the time complexity of generating solutions for larger to-do lists. One step in this direction could be to allow our method to quickly update the point values when the user makes a minor change to its inputs. Additionally, helping users provide more accurate time estimates could allow our method to make more realistic recommendations.

It has been argued that motivating people through extrinsic rewards (rewarding) tends to be harmful in the long run [17]. However, points can be much more than just rewards [24] and gamification is much more than adding points [13]. Our method should therefore be used to design meaningful games that foster intrinsic motivation [17] rather than to just choose the values of extrinsic rewards. A simple first step could be to work out how the task values computed by our method can be conveyed through other game elements, such as levels and badges. This can, in principle, be accomplished through simple rules that specify how many points a user has to earn to attain each level or earn a certain badge.

ACKNOWLEDGMENTS

This work was supported by grant number 1757269 from the National Science Foundation.

REFERENCES

[1] Yossi Aviv and Amit Pazgal. 2005. A partially observed Markov decision process for dynamic pricing. *Management science* 51, 9 (2005), 1400–1416. https://doi.org/10.1287/mnsc.1050.0393

[2] Richard Bellman. 1957. A Markovian decision process. *Journal of mathematics and mechanics* 6, 5 (1957), 679–684.

[3] Shalabh Bhatnagar, Emmanuel Fernández-Gaucherand, Michael C Fu, Ying He, and Steven I Marcus. 1999. A Markov decision process model for capacity expansion and allocation. In *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No. 99CH36304)*, Vol. 2. IEEE, Phoenix, AZ, USA, 1380–1385. https://doi.org/doi={https://doi.org/10.1109/CDC.1999.830146}

[4] Saksham Consul, Jugoslav Stojcheski, Valkyrie Felso, and Falk Lieder. 2021. *Optimal To-Do List Gamification for Long Term Planning.* Technical Report. Max Planck Institute for Intelligent Systems, Tübingen. 30 pages.

[5] Sarah Diefenbach and Annemarie Müssig. 2019. Counterproductive effects of gamification: An analysis on the example of the gamified task manager Habitica. *International Journal of Human-Computer Studies* 127 (2019), 190–210. https://doi.org/10.1016/j.ijhcs.2018.09.004

[6] Adam Maria Gadomski, Sandro Bologna, Giovanni Di Costanzo, Anna Perini, and Marco Schaerf. 2001. Towards intelligent decision support systems for emergency managers: the IDA approach. *International Journal of Risk Assessment and Management* 2, 3-4 (2001), 224–242. https://doi.org/10.1504%2FIJRAM.2001.001507

[7] Ronald A Howard. 1963. Semi-Markovian decision-processes. *Bulletin of the International Statistical Institute* 40, 2 (1963), 625–652.

[8] Daniel Kahneman and Amos Tversky. 1977. *Intuitive prediction: Biases and corrective procedures.* Technical Report. Decisions and Designs Inc Mclean Va.

[9] William R King and Talmadge A Wilson. 1967. Subjective time estimates in critical path planning—a preliminary analysis. *Management science* 13, 5 (1967), 307–320. https://doi.org/10.1287/mnsc.13.5.307

[10] Falk Lieder, Owen X Chen, Paul M Krueger, and Thomas L Griffiths. 2019. Cognitive prostheses for goal achievement. *Nature human behaviour* 3, 10 (2019), 1096–1106. https://doi.org/10.1038/s41562-019-0672-9

[11] Falk Lieder and Tom Griffiths. 2016. Helping people make better decisions using optimal gamification.. In *CogSci*. Cognitive Science Society, Philadelphia, Pennsylvania, 2075–2080.

[12] Brian R Little, Katariina Salmela-Aro, and Susan D Phillips. 2017. *Personal project pursuit: Goals, action, and human flourishing.* Psychology Press, Washington DC, USA.

[13] Athanasios Mazarakis. 2021. Gamification Reloaded. *i-com* 20, 3 (2021), 279–294. https://doi.org/10.1515/icom-2021-0025

[14] Elisa D Mekler, Florian Brühlmann, Klaus Opwis, and Alexandre N Tuch. 2013. Do points, levels and leaderboards harm intrinsic motivation? An empirical analysis of common gamification elements. In *Proceedings of the First International Conference on gameful design, research, and applications.* ACM, Stratford, Canada, 66–73. https://doi.org/10.1145/2583008.2583017

[15] Elisa D Mekler, Florian Brühlmann, Alexandre N Tuch, and Klaus Opwis. 2017. Towards understanding the effects of individual gamification elements on intrinsic motivation and performance. *Computers in Human Behavior* 71 (2017), 525–534. https://doi.org/10.1016/j.chb.2015.08.048

[16] Kou Murayama. 2022. A reward-learning framework of knowledge acquisition: An integrated account of curiosity, interest, and intrinsic–extrinsic rewards. *Psychological Review* 129, 1 (2022), 175. https://doi.org/10.1037/rev0000349

[17] Scott Nicholson. 2015. A recipe for meaningful gamification. In *Gamification in education and business.* Springer, Switzerland, 1–20. https://doi.org/10.1007/978-3-319-10208-5_1

[18] Luiz Guilherme Nadal Nunes, Solon Venancio de Carvalho, and Rita de Cássia Meneses Rodrigues. 2009. Markov decision process applied to the control of hospital elective admissions. *Artificial intelligence in medicine* 47, 2 (2009), 159–171. https://doi.org/10.1016/j.artmed.2009.07.003

[19] Haili Song, C-C Liu, Jacques Lawarrée, and Robert W Dahlgren. 2000. Optimal electricity supply bidding by Markov decision process. *IEEE transactions on power systems* 15, 2 (2000), 618–624. https://doi.org/10.1109/59.867150

[20] Piers Steel. 2007. The nature of procrastination: a meta-analytic and theoretical review of quintessential self-regulatory failure. *Psychological bulletin* 133, 1 (2007), 65. https://doi.org/10.1037/0033-2909.133.1.65

[21] Piers Steel and Cornelius J König. 2006. Integrating theories of motivation. *Academy of management review* 31, 4 (2006), 889–913. https://doi.org/10.5465/amr.2006.22527462

[22] Jugoslav Stojcheski, Valkyrie Felso, and Falk Lieder. 2020. *Optimal to-do list gamification.* Technical Report. Max Planck Institute for Intelligent Systems, Tübingen. 29 pages.

[23] Armando M Toda, Pedro HD Valle, and Seiji Isotani. 2017. The dark side of gamification: An overview of negative effects of gamification in education. In *Researcher links workshop: higher education for all.* Springer, Springer, Switzerland, 143–156. https://doi.org/10.1007/978-3-319-97934-2_9

[24] Kevin Werbach and Dan Hunter. 2020. *For the Win, Revised and Updated Edition: The Power of Gamification and Game Thinking in Business, Education, Government, and Social Impact.* University of Pennsylvania Press, Philadelphia, PA, USA.