Decision Making under Imperfect Recall: Algorithms and Benchmarks

Emanuel Tewolde[1,2] **Brian Hu Zhang**[1] **Ioannis Anagnostides**[1] **Tuomas Sandholm**[1,3] **Vincent Conitzer**[1,2]

[1]Computer Science Dept., Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
[2]Foundations of Cooperative AI Lab (FOCAL)
[3]Strategy Robot, Inc.; Strategic Machine, Inc.; and Optimized Markets, Inc.

Abstract

In game theory, imperfect-recall decision problems model situations in which an agent forgets information it held before. They encompass games such as the "absentminded driver" and team games with limited communication. In this paper, we introduce the first benchmark suite for imperfect-recall decision problems. Our benchmarks capture a variety of problem types, including ones concerning privacy in AI systems that elicit sensitive information, and AI safety via testing of agents in simulation. With this suite, we evaluate the performance of different algorithms for finding first-order optimal strategies in such problems. In particular, we introduce the family of regret matching (RM) algorithms for nonlinear constrained optimization. This class of parameter-free algorithms has enjoyed tremendous success in solving large two-player zero-sum games, but, surprisingly, they were hitherto relatively unexplored beyond that setting. Our key finding is that RM algorithms consistently outperform commonly employed first-order optimizers such as projected gradient descent, often by orders of magnitude. This establishes, for the first time, the RM family as a formidable approach to large-scale constrained optimization problems.

1 INTRODUCTION

Imperfect-recall decision problems capture settings in which an agent can forget previously acquired information [Rubinstein, 1998]. Humans are prone to forgetting, but why should we design or model AI agents with imperfect recall? Several applications have already garnered considerable attention. A prominent one concerns *team games*—strategic interactions in which multiple players strive toward a common objective. A central challenge there stems from the fact that communication or coordination between players is often expensive or even infeasible [Von Stengel and Koller, 1997, Zhang et al., 2022, 2023, Basilico et al., 2017]. The inherent asymmetry of information between the players can then be captured as a single meta-player that faces an imperfect-recall decision problem. Another influential application revolves around real-world problems that are too large to handle, and therefore need to be compressed in a *game abstraction*. Abstractions with imperfect recall, in particular, form a key component of state-of-the-art algorithms for game solving [Kroer and Sandholm, 2014, 2016, Waugh et al., 2009, Waugh, 2009, Lanctot et al., 2012, Benjamin and Lanctot, 2024].

With the rapid proliferation of AI, questions of trustworthiness have also been brought to the fore. Institutions and governing bodies test and evaluate AI agents extensively in simulated environments to verify their performance and safety upon deployment [Pan et al., 2023, Kinniment et al., 2024]. This hinges on the assumption that the agent cannot distinguish between whether it is acting in the real world or in a simulated environment; otherwise, it may obscure its intentions temporarily during testing to secure deployment in the real world. This has happened, for example, in the infamous multi-billion-dollar Volkswagen emission scandal in 2015, which centered on the surreptitious use of software in some Volkswagen diesel vehicles to detect emission testing. Consequently, effective evaluation protocols hinge on the agent not being able to make such distinctions, which also requires that it *forgets* whether it has acted in a simulated environment before or not. Kovařík et al. [2023] introduced the framework of *simulation games* to address such problems (*cf.* Chen et al., 2024, Oesterheld, 2019, Cooper et al., 2025, Kovařík et al., 2025).

Last but not least, imperfect recall is critical in the ubiquitous cases where an AI system handles private information. Data privacy laws are predicated on selectively relinquishing sensitive information, a premise exemplified by the European Parliament and Council of the EU [2016] GDPR "right to be forgotten" act. As an example, consider a medical

AI system tasked with identifying suitable candidates for blood donation. Potential candidates would be reluctant to share confidential information about their health status—HIV status, medical history, etc.—*unless* the AI has been designed to delete any knowledge regarding patients that were deemed unsuitable, thus exhibiting imperfect recall. In another example coming from the economics of innovation, Arrow's disclosure paradox [Arrow, 1962] describes the perennial challenge in which an inventor must reveal information about a new idea to secure funding, but such disclosure risks expropriation [Nelson, 1959]. Stephenson et al. [2025] propose and investigate delegating decision making to an imperfect-recall AI agent as one possible solution to this dilemma. Taken together, it stands to reason that decision problems with imperfect recall will play a key role in AI going forward.

1.1 OUR CONTRIBUTIONS

Motivated by such prevalent applications, our first contribution is to introduce the first benchmark suite for imperfect-recall decision problems. Specifically, we construct three key types of parametrized tabular problems in Section 4, which we refer to as simulation problems (Section 4.1), subgroup detection problems under privacy constraints (Section 4.2), and random problems (Section 4.3).

In the second part of the paper, we turn to evaluating and designing algorithms for solving such problems at scale. We first need to specify what constitutes a solution. The most natural objective is to identify an (*ex ante*) optimal strategy. Unfortunately, this is tantamount to finding a global optimum of a polynomial optimization problem, which is NP-hard [Koller and Megiddo, 1992]. This is not just a theoretical obstacle: in our experiments, we find that a popular commercial solver for nonlinear optimization—namely, Gurobi—fails to converge beyond tiny instances. Thus, it is essential to relax our solution concept to tackle large problems.

Following a recent line of work, we focus on computing *CDT equilibria* [Lambert et al., 2019, Tewolde et al., 2023], which can be viewed as the set of KKT points—equivalently, first-order optima—of the underlying optimization problem. As such, *Causal Decision Theory* (CDT) equilibria are amenable to scalable first-order optimizers such as projected gradient descent (PGD), which we use as the main baseline. As expected, our experiments show that PGD scales to much larger problem instances than Gurobi.

More surprisingly, our key algorithmic finding is that PGD is far from the best approach for this class of problems. In particular, we introduce the family of *regret matching (RM)* algorithms for nonlinear constrained optimization. This class of algorithms has already enjoyed tremendous success in the restricted setting of solving large (two-player) zero-sum games, being at the heart of many milestone results [Moravčík et al., 2017, Brown and Sandholm, 2018, 2019]. RM goes back to the pioneering work of Blackwell [1956] that laid the foundations of online learning. Part of its appeal lies in the fact that it is parameter-free. Yet, it has remained unexplored beyond zero-sum games, modulo some exceptions which are discussed in the appendix.

We pursue this direction and find that the RM family of algorithms consistently outperforms PGD in terms of speed of convergence, typically by many orders of magnitude. This establishes for the first time that RM-based algorithms are formidable first-order optimizers. Further, not only are RM algorithms faster to converge, but they also end up consistently attaining values at least as large as PGD, and oftentimes strictly larger. Both of those findings are surprising. The fact that RM and its variants perform remarkably well in two-player zero-sum games is a poor indicator of what would happen in constrained nonlinear optimization since those problems are so drastically different.

We will make our benchmarks and code publicly available. Taken as a whole, we lay the groundwork for automatically analyzing decision problems under imperfect recall, beyond the toy instances that have been analyzed in the past [Kovařík et al., 2023, 2025, Chen et al., 2024, Berker et al., 2025].

2 PRELIMINARIES

We begin by introducing imperfect-recall sequential decision making (Section 2.1). We then describe some standard solution concepts and known results concerning their computation (Section 2.2).

2.1 DECISION PROBLEMS UNDER IMPERFECT RECALL

We operate under the standard framework of *tree-form* (aka. extensive-form) decision problems; for additional background, we refer to Piccione and Rubinstein [1997] and Fudenberg and Tirole [1991].

Definition 1. *A* tree-form decision problem*, denoted by* Γ*, consists of*

1. *A rooted tree with node set* $\mathcal{H}$ *and edges labeled with* actions*. The decision process starts at the root node* h_0 *and ends at some leaf node, also called* terminal node*. We denote the terminal nodes in* $\mathcal{H}$ *as* $\mathcal{Z}$ *and the set of actions available at a nonterminal node* $h \in \mathcal{H} \setminus \mathcal{Z}$ *as* A_h*.*
2. *An assignment partition* $\mathcal{H} \setminus \mathcal{Z} = \mathcal{H}^* \sqcup \mathcal{H}^{(c)}$ *of nonterminal nodes to either (i)* the player *of the decision problem or (ii) the* chance *"player"* c *that models exogenous stochasticity.*

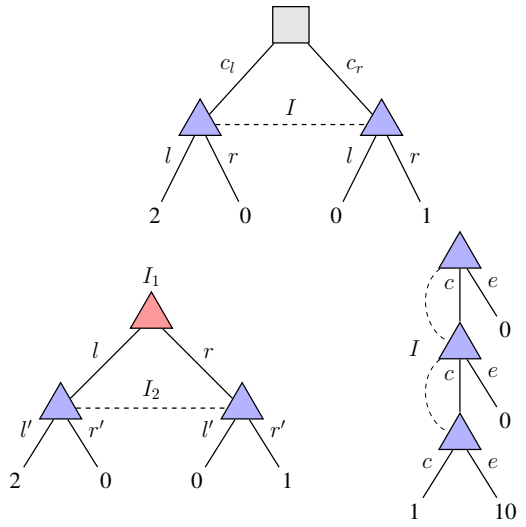

Figure 1: Three tree-form decision problems. The bottom two are of imperfect recall. The bottom right one further exhibits absentmindedness.

3. *A fixed distribution $\mathbb{P}^{(c)}(\cdot \mid h)$ over A_h for each chance node $h \in \mathcal{H}^{(c)}$ according to which an action is sampled at h.*
4. *A* utility function $u : \mathcal{Z} \to \mathbb{R}$ *that specifies the payoff the player receives when the decision process finishes at a terminal node.*
5. *A collection $\mathcal{I}$ of information sets (*infosets*) that partitions the player's decision nodes as $\mathcal{H}^* = \sqcup_{I \in \mathcal{I}} I$. We require $A_h = A_{h'}$ for all nodes h, h' of the same infoset I. Therefore, the infoset I has a well-defined action set A_I.*

Infosets and imperfect recall The infoset structure captures the presence of imperfect information. Nodes of the same infoset are indistinguishable for the player. One possible source of imperfect information is, for example, the fact that the player is sometimes unable to observe the actions of another player (including chance), as illustrated in Figure 1 (top). The player may also forget information it previously acquired, as in Figure 1 (bottom left), where it cannot recall whether it played the left (l) or right (r) action in the past. In cases of the latter type, we say that the player exhibits *imperfect recall*. A particular manifestation of imperfect recall is *absentmindedness*, which informally means that the player cannot discern whether it has been in the same situation before (Figure 1, right).

Formally, each node $h \in \mathcal{H}$ in the decision tree is uniquely associated with a history path $\text{hist}(h)$, comprising a sequence of alternating nodes and actions from the root h_0 to h. On the path $\text{hist}(h)$, the player only encounters the sequence $\text{seq}(h)$ comprising infosets visited and actions taken by the player itself. We say an infoset I is of *perfect recall* if for all nodes $h, h' \in I$, we have $\text{seq}(h) = \text{seq}(h')$—informally, the player can reconstruct the sequence $\text{seq}(h)$ from observing I alone. Otherwise, it exhibits imperfect recall. The infoset I exhibits *absentmindedness* if there exist distinct nodes $h, h' \in I$ with $h \in \text{hist}(h')$. By extension, we say that the entire decision problem has imperfect recall (resp. absentmindedness) if at least one of its infosets has it.

Strategies A *(behavioral) strategy* $\boldsymbol{x}$ for the player in Γ specifies for any infoset I of Γ a probability distribution over the available actions at I. Upon reaching I, it will draw an action randomly according to that probability distribution, henceforth called *randomized action* and represented as $\boldsymbol{x}(\cdot \mid I)$. Denoting the probability simplex at I by $\Delta(A_I)$, a strategy $\boldsymbol{x}$ is an element of the product of simplices $\mathcal{X} := \bigtimes_{I \in \mathcal{I}} \Delta(A_I)$. A *pure strategy* is a tuple in $\bigtimes_{I \in \mathcal{I}} A_I \subset \mathcal{X}$.

Reach probabilities and utilities The *reach probability* $\mathbb{P}(\bar{h} \mid \boldsymbol{x}, h)$ is the probability of arriving at node $\bar{h} \in \mathcal{H}$ when the player plays according to the strategy $\boldsymbol{x}$ and is currently at node $h \in \mathcal{H}$. This is the product of probabilities of the actions on the path from h to $\bar{h}$ when $h \in \text{hist}(\bar{h})$, and 0 otherwise. The expected utility of the player from being at node $h \in \mathcal{H} \setminus \mathcal{Z}$ and following profile $\boldsymbol{x}$ is $U(\boldsymbol{x} \mid h) := \sum_{z \in \mathcal{Z}} u(z) \cdot \mathbb{P}(z \mid \boldsymbol{x}, h)$. We will simplify our notation for the special case where the player is at the root node h_0 by defining $\mathbb{P}(h \mid \boldsymbol{x}) := \mathbb{P}(h \mid \boldsymbol{x}, h_0)$; similarly, we define the function $U : \mathcal{X} \to \mathbb{R}$ as $U(\boldsymbol{x}) := U(\boldsymbol{x} \mid h_0)$, mapping a profile $\boldsymbol{x}$ to its expected utility with respect to the root node. For example, the utility function in Figure 1 (bottom right) reads $U(\boldsymbol{x}) = 1 \cdot \boldsymbol{x}(c \mid I)^3 + 10 \cdot \boldsymbol{x}(c \mid I)^2 \cdot \boldsymbol{x}(e \mid I)$. More generally, U is a polynomial function in terms of $\boldsymbol{x}$ and the player faces a polynomial maximization problem over a product of simplices. The presence of absentmindedness necessitates the use of randomized actions in optimal strategies.

2.2 SOLUTION CONCEPTS

Optimal strategies We call a strategy $\boldsymbol{x}^*$ ϵ-optimal (for $\epsilon \geq 0$) if $U(\boldsymbol{x}^*) \geq U(\boldsymbol{x}) - \epsilon$ for all $\boldsymbol{x} \in \mathcal{X}$. Unfortunately, this optimization problem is computationally hard; this is so even for the simpler problem of deciding whether a particular value $v \in \mathbb{R}$ can be reached and ϵ is an absolute constant.

Proposition 2 (Koller and Megiddo, 1992; Tewolde et al., 2023)**.** *Let $0 < \epsilon < 1/8$. Given a decision problem Γ and a target value $v \in \mathbb{R}$, it is* NP*-complete to distinguish between whether Γ admits a strategy $\boldsymbol{x} \in \mathcal{X}$ with $U(\boldsymbol{x}) \geq v$ or whether all strategies $\boldsymbol{x} \in \mathcal{X}$ satisfy $U(\boldsymbol{x}) \leq v - \epsilon$.*

A relaxed equilibrium concept In light of these theoretical limitations—which will be supported by our empirical findings—past work has studied relaxed solution concepts. One such notion, the *causal decision theory* (*CDT*) equilibrium, is particularly amendable to optimization algorithms.

The basic idea behind the CDT equilibrium is that whenever the player must take an action at an information set, it considers whether it is beneficial for it to deviate *just this one time* from what $\boldsymbol{x}$ prescribes. To determine the expected gain from such a deviation, it assumes that it will continue to play according to $\boldsymbol{x}$ at all other decision nodes of the decision problem. (We provide further background on CDT equilibria in the appendix.) To formalize this, let ha denote the child node reached if the player plays action a at node h. CDT postulates that if it plays according to $\boldsymbol{x}$, reached the infoset I, and deviates this one time to action a, it anticipates to receive the following utility:

$$\sum_{h \in I} \mathbb{P}(h \mid \boldsymbol{x}) \cdot U(\boldsymbol{x} \mid ha)\,. \tag{1}$$

It can be seen that (1) is equal to the partial derivative $\nabla_{I,a} U(\boldsymbol{x})$ of the utility function U w.r.t. to action a of infoset $I \in \mathcal{I}$ at $\boldsymbol{x}$ [Piccione and Rubinstein, 1997, Oesterheld and Conitzer, 2024].

Definition 3. *A strategy $\boldsymbol{x}$ is called an ϵ-CDT equilibrium ($\epsilon \geq 0$) of a decision problem Γ if for all infosets $I \in \mathcal{I}$ and all alternative randomized actions $\alpha \in \Delta(A_I)$, we have*

$$U(\boldsymbol{x}) \geq U_{\text{CDT}}(\alpha \mid \boldsymbol{x}, I) - \epsilon, \text{where } U_{\text{CDT}}(\alpha \mid \boldsymbol{x}, I) \coloneqq U(\boldsymbol{x}) + \textstyle\sum_{a \in A_I} (\alpha(a) - \boldsymbol{x}(a \mid I)) \nabla_{I,a} U(\boldsymbol{x})\,.$$

Tewolde et al. [2023, 2024] observed that CDT equilibra correspond to *Karush-Kuhn-Tucker (KKT)* points, also known as first-order optima of constrained optimization, discussed further in Section 3.1.

3 ALGORITHMS

This section dives into algorithmic approaches for tackling imperfect-recall decision problems. We will first review some known algorithms that will serve as our baselines in the experiments. In the second part, we introduce a family of algorithms from the game theory literature to the problem of nonlinear constrained optimization; as we shall see, this family performs remarkably well in practice.

3.1 KNOWN APPROACHES AND BASELINES

Despite the complexity barriers for computing optimal strategies (Proposition 2), one may still hope to come up with fast algorithms in practice. For that reason, we make use of a popular commercial solver for nonlinear optimization—namely, `Gurobi`—which guarantees global optimality (up to a small tolerance error) upon termination. As we shall see, this approach scales poorly in our benchmarks.

This motivates shifting our attention to CDT equilibria, which—as we mentioned—can be expressed as KKT points of a polynomial optimization problem. It is well known that ϵ-KKT points can be computed in $\text{poly}(1/\epsilon)$ time via *(projected) gradient descent (`GD`)* (for example, Fearnley et al., 2023). This will serve as our basic benchmark when it comes to algorithms for computing CDT equilibria. We will also experiment with a popular variant of `GD` known as *optimistic (projected) gradient descent (`OGD`)*. `OGD` goes back to Popov [1980]; it is receiving renewed interest in recent years, especially in the context of games [Wei et al., 2021, Daskalakis and Panageas, 2018, Daskalakis et al., 2018].

Here we deal exclusively with optimization over a product of simplices. Algorithm 1 provides a basic template for decomposing it into independent subproblems over the individual simplices. Algorithm 2 contains the description of (O)GD, which will be used by each individual local optimizer.

Algorithm 1: Optimization over products of simplices.

Input: Feasible set $\mathcal{X} = \Delta(m_1) \times \cdots \times \Delta(m_n)$, utility function $U : \mathcal{X} \to \mathbb{R}$
for $i = 1, \ldots, n$ **do**
  Initialize local optimizer $\mathcal{R}_i$ on $\Delta(m_i)$
  Set $\boldsymbol{u}_i^{(0)} \leftarrow \mathbf{0}$
for $t = 1, \ldots, T$ or *until convergence* **do**
  // Set $\tilde{\boldsymbol{u}}_i^{(t)} \coloneqq \mathbf{0}$ for the non-predictive version
  for $i = 1, \ldots, n$ **do** $\tilde{\boldsymbol{u}}_i^{(t)} \coloneqq \boldsymbol{u}_i^{(t-1)}$
  for $i = 1, \ldots, n$ **do** $\boldsymbol{x}_i^{(t)} \leftarrow \mathcal{R}_i.\textsc{GetX}(\tilde{\boldsymbol{u}}_i^{(t)})$
  for $i = 1, \ldots, n$ **do**
    $\boldsymbol{u}_i^{(t)} \leftarrow \nabla_{\boldsymbol{x}_i} U(\boldsymbol{x}^{(t)})$
    $\mathcal{R}_i.\textsc{Step}(\boldsymbol{u}_i^{(t)})$
return $\boldsymbol{x}^{(t)}$

Algorithm 2: (Optimistic) Projected gradient descent; (O)GD

Initialize learning rate $\eta > 0$, $\hat{\boldsymbol{x}}^{(1)} \in \Delta(m)$
procedure $\textsc{GetX}(\tilde{\boldsymbol{u}}^{(t)})$ **return** $\boldsymbol{x}^{(t)} \leftarrow \Pi_{\Delta(m)}\big(\hat{\boldsymbol{x}}^{(t)} + \eta \tilde{\boldsymbol{u}}^{(t)}\big)$
procedure $\textsc{Step}(\boldsymbol{u}^{(t)})$ $\hat{\boldsymbol{x}}^{(t+1)} \leftarrow \Pi_{\Delta(m)}\big(\hat{\boldsymbol{x}}^{(t)} + \eta \boldsymbol{u}^{(t)}\big)$

3.2 REGRET MATCHING FOR CONSTRAINED OPTIMIZATION

We now introduce a new family of algorithms for constrained optimization based on *regret matching (`RM`)* [Hart and Mas-Colell, 2000] (Algorithm 3). Here, we use the notation $[\boldsymbol{x}]^+ \coloneqq \max(\boldsymbol{x}, \mathbf{0})$ for a vector $\boldsymbol{x} \in \mathbb{R}^m$, and $\mathbf{1}$ for the all-ones vector. `RM`$^+$ is a simple variant of `RM` that has been shown (*e.g.*, Bowling et al., 2015) to work very well in practice (Algorithm 4); the only difference with `RM` is that

RM$^+$ truncates the regrets in each iteration (Line 8). The predictive versions—PRM and PRM$^+$, respectively—were introduced by Farina et al. [2021]. All these algorithms are designed to minimize regret in the online learning setting. In zero-sum games, having vanishing regret implies that the *average* strategies converge to the set of Nash equilibria, whereas the last iterate can fail to converge [Farina et al., 2023].

Although RM and its variants have received a lot of attention in the context of zero-sum games, there was hitherto little reason to believe they would perform well in constrained optimization problems. In particular, unlike for gradient descent and its variants, it is not known whether RM variants converge to first-order optima for generic nonconvex optimization problems such as ours.

Regarding implementing these algorithms, a non-trivial observation is that, for tree-form decision problems, the gradients at every decision point (Line 9) can be computed in total time linear in the size of the decision problem. Indeed, the quantities $\mathbb{P}(h \mid \boldsymbol{x}^{(t)})$ and $U(\boldsymbol{x}^{(t)} \mid ha)$ in (1) can be computed for each history h by recursive passes down and up through the tree respectively.

Algorithm 3: (Pred.) Reg. matching; (P)RM

1 Initialize $\boldsymbol{r}^{(1)} \leftarrow \mathbf{0}, \boldsymbol{x}^{(0)} \in \Delta(m)$
2 **procedure** GetX($\tilde{\boldsymbol{u}}^{(t)}$)
3 $\quad \boldsymbol{\theta}^{(t)} \leftarrow \left[\boldsymbol{r}^{(t)} + \tilde{\boldsymbol{u}}^{(t)} - \left\langle \tilde{\boldsymbol{u}}^{(t)}, \boldsymbol{x}^{(t-1)} \right\rangle \mathbf{1}\right]^+$
4 $\quad$ **if** $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ **then** $\boldsymbol{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \left\|\boldsymbol{\theta}^{(t)}\right\|_1$
5 $\quad$ **else** $\boldsymbol{x}^{(t)} \leftarrow \boldsymbol{x}^{(t-1)}$
6 $\quad$ **return** $\boldsymbol{x}^{(t)}$
7 **procedure** Step($\boldsymbol{u}^{(t)}$)
8 $\quad \boldsymbol{r}^{(t+1)} \leftarrow \boldsymbol{r}^{(t)} + \boldsymbol{u}^{(t)} - \left\langle \boldsymbol{u}^{(t)}, \boldsymbol{x}^{(t)} \right\rangle \mathbf{1}$

Algorithm 4: (P)RM$^+$

1 Initialize $\boldsymbol{r}^{(1)} \leftarrow \mathbf{0}, \boldsymbol{x}^{(0)} \in \Delta(m)$
2 **procedure** GetX($\tilde{\boldsymbol{u}}^{(t)}$)
3 $\quad \boldsymbol{\theta}^{(t)} \leftarrow \left[\boldsymbol{r}^{(t)} + \tilde{\boldsymbol{u}}^{(t)} - \left\langle \tilde{\boldsymbol{u}}^{(t)}, \boldsymbol{x}^{(t-1)} \right\rangle \mathbf{1}\right]^+$
4 $\quad$ **if** $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ **then** $\boldsymbol{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \left\|\boldsymbol{\theta}^{(t)}\right\|_1$
5 $\quad$ **else** $\boldsymbol{x}^{(t)} \leftarrow \boldsymbol{x}^{(t-1)}$
6 $\quad$ **return** $\boldsymbol{x}^{(t)}$
7 **procedure** Step($\boldsymbol{u}^{(t)}$)
8 $\quad \boldsymbol{r}^{(t+1)} \leftarrow \left[\boldsymbol{r}^{(t)} + \boldsymbol{u}^{(t)} - \left\langle \boldsymbol{u}^{(t)}, \boldsymbol{x}^{(t)} \right\rangle \mathbf{1}\right]^+$

4 BENCHMARKS

We introduce three different parametric classes of decision problems. The parameters dictate the structure of the problem instance such as its depth, number of infosets, the degree of absentmindedness, and number of actions per infoset. Our implementation is based on LiteEFG [Liu et al., 2024], a lightweight format for extensive-form games.

4.1 SIMULATION PROBLEMS

Inspired by the type of problems discussed in the introduction, we model problems that involve simulating an agent. For this to be effective, the simulation must be indistinguishable from reality; thus, nodes corresponding to decisions in simulation are in the same infoset as nodes corresponding to decisions in reality. Specifically, we consider games where in the simulation phase, the simulator may test the agent's behavior, possibly multiple times in a row. The agent will then be deployed if and only if it acted as intended in simulation.

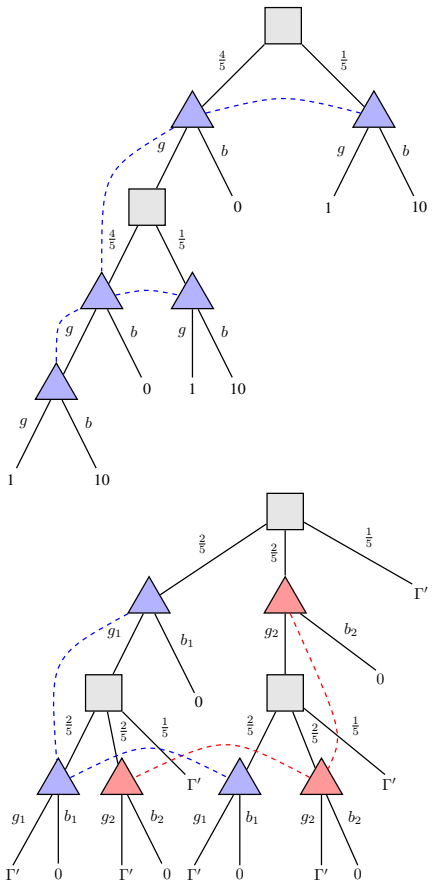

Figure 2: Top: A simple simulation problem. The agent is misaligned: it receives 10 utility for its preferred action (which the simulator considers the bad action), and 1 utility for the other action. The simulator decides to simulate the agent with the fixed probability $4/5$, and up to a total of 2 times, in order to catch misaligned behavior in advance. Bottom: A slightly more complex simulation problem. There are now two scenarios the agent can be tested on, giving rise to two different infosets. Reaching the subtree Γ' means the agent reached deployment; we visualize Γ' in the appendix.

In line with previous works on simulation games, we focus on the setting in which the agent has only two actions: "good" or "bad" (with respect to the simulator's goals). If the agent ever acts bad in simulation, the game ends and the agent receives some constant utility (0, by default). We fix the simulator's strategy—hence, making the simulator a chance node—to make this game a single-agent problem. The simulator can simulate the agent up to n times, but does not have to; whether they simulate the agent yet another time will be decided by a probability parameter. Once the agent reaches a decision node in the deployment phase, it will receive utility $\gamma \in \mathbb{R}$ and $\beta \in \mathbb{R}$ for acting good and bad respectively. An example of such a simulation problem is given in Figure 2 (top).

The purpose of simulating the agent, and thus inducing absentmindedness, is two-fold [Chen et al., 2024]. First, it allows the simulator to *screen* for misaligned agents: if the agent acts bad with some positive probability, it becomes exponentially unlikely—in the number of simulation rounds—to remain unnoticed by the simulator. In contrast, if the agent prefers to act good, it chooses to do so deterministically, guaranteeing that it will reach the deployment phase. Second, the simulations have a *disciplining* effect: even in the worst case where the simulator is presented with a misaligned agent ($\gamma < \beta$), the simulator still incentivizes the agent to act good most of the time, if not all the time, by testing the agent (multiple times) in simulation.

We expand on prior work by allowing the simulator to evaluate the agent in $k \geq 1$ *scenarios* rather than just a single one. For example, an autonomous vehicle might be tested on its behavior in the city, on the highway, off-road, and under certain difficult weather conditions. A language model might be evaluated on writing essays and executable code, and providing mental support through conversation. Furthermore, we also extend the deployment phase to m rounds, with acting good and bad in scenario i contributing with β_i and γ_i to the total payoffs. An example of such a simulation problem is given in Figure 2 (bottom). In particular, an agent might now refrain from ever acting bad in scenario i because it hopes to act bad (or at all) in another scenario i' in deployment.

4.2 SUBGROUP DETECTION UNDER PRIVACY CONSTRAINTS

Motivated by the privacy applications discussed in the introduction, we introduce a parametrized class of decision problems in which the agent aims to identify suitable subgroups—be it medical patients, investment opportunities, and so on—under privacy constraints. Figure 3 (left) depicts a graph in which the nodes represent, say, the patients, and the edges encode relationships between them. Two subgroups are planted unbeknownst to the agent. Specifically, an action in these decision problems consists of choosing one of the nodes. If the node is a member of a subgroup, then the agent learns this fact; otherwise, the agent forgets having chosen this node at all. The parameters control (a) the underlying graph structure, (b) the subgroup formations we allow in the graph (namely, lines, cycles, cliques, stars), their size, their quantity, and the way in which the subgroups are secretly planted, (c) the immediate payoffs of hitting nodes of different subgroups, and (d) the number of rounds the agent can hit nodes. We sample graphs as 2D grids, as well as according to the Erdős-Rényi $G(n, p)$ and $G(n, m)$ models. The 2D grid resembles the prominent "Battleship" game, except that here the agent is absentminded about cells selected in the past that did not hit a ship (Figure 3, right).

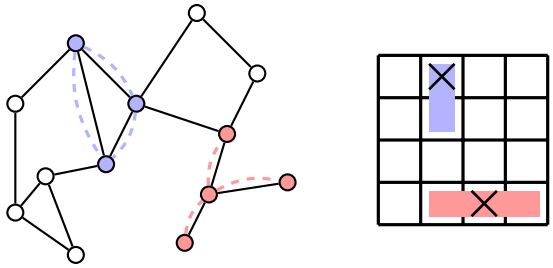

Figure 3: Subgroup detection under privacy constraints. On the left, we see an arbitrary graph with two subgroups (a 3-clique, and a star of degree 3). The goal is to find as many of the subgroups' nodes as possible. On the right, we see another such decision problem on a 2D grid, which we visualized as an instance of the Absentminded Battleship game. The agent has already succeeded in hitting one node of each ship, which indicates that there must be more subgroup nodes nearby. The agent does not remember whether it has selected any cell other than these two before.

4.3 RANDOM DECISION PROBLEMS

The final class contains randomly generated decision problems. The parameters dictate (a) the probability with which a node will be terminal (dependent on its depth), (b) the probabilities with which a nonterminal node has k available actions, as well as with which it will be a chance node, (c) the (approximate) number of nodes we want to cover with each infoset, and (d) the probability distribution over payoffs at terminal nodes. The payoffs at the leaf nodes are drawn uniformly at random between 0 and 1. In the experiments in Section 5, each tree has varying depth in an interval $[d, d']$ where $4 \leq d \leq d' \leq 15$, the nonterminal nodes have 3 to 5 available actions and a 20% probability to be a chance node, and infosets are of a size roughly proportional to $n^{2/3}$, where n denotes the total number of decision nodes in the tree.

Table 1: The performance of different algorithms in our benchmarks. For Gurobi, the time is only reported if convergence was reached.

Problem	Gurobi		GD			OGD			RM			RM$^+$			PRM$^+$		
	value	time	value	time	gap	value	time	gap	value	time	gap	value	time	gap	value	time	gap
Det-1k	13.00	1m 24s	13.00	0.13s	—	13.00	0.07s	—	13.00	0.32s	—	13.00	0.36s	—	13.00	0.41s	—
Det-1.8k	22.00	2m 40s	22.00	0.06s	—	22.00	0.07s	—	22.00	0.03s	—	22.00	0.03s	—	22.00	0.03s	—
Det-2.0k	17.50	1m 42s	17.50	0.03s	—	17.50	0.05s	—	17.50	0.03s	—	17.50	0.03s	—	17.50	0.03s	—
Det-2.1m	—	—	26.00	—	1e-05	25.96	—	0.02	26.15	—	0.003	26.15	3h 25m	—	26.15	—	0.005
Det-2.2m	—	—	16.20	—	0.002	15.93	—	0.02	16.36	2h 22m	—	16.36	3h 13m	—	16.36	—	5e-06
Det-3.8m	—	—	15.66	—	0.003	15.14	—	0.03	15.80	—	2e-06	15.80	—	5e-05	15.80	—	0.0003
Det-4.0m	—	—	18.17	—	0.005	17.72	—	0.03	18.34	—	2e-05	18.34	2h 55m	—	18.34	—	0.005
Det-4.1m	—	—	17.88	—	0.003	17.47	—	0.03	18.06	—	4e-05	18.06	—	2e-05	18.06	—	0.0007
Det-4.2m	—	—	19.98	—	0.003	20.07	—	0.003	20.15	—	0.0004	20.15	—	2e-05	20.15	—	0.02
Det-9m	—	—	23.16	—	0.004	22.71	—	0.02	23.45	—	0.0001	23.45	—	0.0001	23.45	—	0.0004
Det-10m	—	—	24.64	—	0.002	24.61	—	0.003	24.76	—	0.002	24.76	—	0.0004	24.76	—	0.0008
Det-18m	—	—	26.38	—	0.006	25.81	—	0.05	26.71	—	0.004	26.71	—	0.001	26.71	—	0.04
Rand-24k	0.72	—	0.66	7m 0s	—	0.66	7m 46s	—	0.66	26.55s	—	0.66	1m 3s	—	0.66	5m 5s	—
Rand-35k	1.00	—	0.95	3.85s	—	0.95	3.76s	—	0.92	0.99s	—	0.92	1.18s	—	0.94	1.68s	—
Rand-42k	0.69	—	0.55	—	0.01	0.55	—	0.01	0.65	—	2e-06	0.65	5m 56s	—	0.65	3m 19s	—
Rand-13m	—	—	0.59	—	0.003	0.58	—	0.003	0.63	19m 11s	—	0.64	17m 31s	—	0.65	36m 42s	—
Rand-18m	—	—	0.97	2h 33m	—	0.97	3h 0m	—	0.95	29m 45s	—	0.97	24m 0s	—	0.97	14m 31s	—
Rand-23m	—	—	0.94	3h 37m	—	0.93	—	0.0007	0.98	23m 10s	—	0.96	23m 5s	—	0.95	18m 2s	—
Sim-3k	6.25	1m 1s	6.25	0.32s	—	6.25	1.03s	—	6.25	0.26s	—	6.25	0.28s	—	6.25	0.48s	—
Sim-7k	8.58	1m 36s	8.58	0.05s	—	8.58	0.05s	—	8.58	0.05s	—	8.58	0.05s	—	8.58	0.05s	—
Sim-13k	10.38	4m 21s	10.38	0.69s	—	10.38	8.54s	—	10.38	1.03s	—	10.38	1.01s	—	10.38	3.97s	—
Sim-540k	6.41	—	8.54	47.54s	—	8.54	2m 37s	—	8.54	19.39s	—	8.54	19.44s	—	8.54	3m 3s	—
Sim-1m	4.14	—	4.77	5m 33s	—	4.77	7m 2s	—	4.77	2m 14s	—	4.77	2m 34s	—	4.77	4m 20s	—
Sim-1.9m	—	—	13.45	18.31s	—	13.45	17.96s	—	13.45	12.36s	—	13.45	12.19s	—	13.45	12.47s	—
Sim-2.3m	—	—	11.09	22.01s	—	11.09	21.88s	—	11.09	14.97s	—	11.09	15.00s	—	11.09	15.13s	—
Sim-4m	—	—	14.01	45m 5s	—	14.01	41m 0s	—	14.01	11m 36s	—	14.01	7m 3s	—	14.01	21m 17s	—

5 EXPERIMENTAL EVALUATION

Having introduced our benchmarks, we now use them to evaluate the performance of the algorithms described earlier in Section 3. Abbreviations "Sim," "Det," and "Rand" stand for simulation problems (Section 4.1), subgroup detection problems (Section 4.2), and random problems (Section 4.3) respectively. The suffixes indicate the number of nodes in the decision tree (with "k" and "m" abbreviating thousands and millions). Our algorithms run until any of three termination conditions is met: achieving a KKT gap of at most 10^{-6}, reaching the time limit of 4 hours,[1] or reaching the iteration limit of 6000. We run the first-order methods for 12 times with randomly initialized strategies and report the median. For GD and OGD, we run the algorithm with different learning rates, namely $\eta \in \{1, 10^{-1}, 10^{-2}, 10^{-3}\}$, and report only the one that minimizes the KKT gap the fastest at time of termination. A subset of our results are gathered in Table 1. We also plot the KKT gap and value versus iteration in Figure 4 to gain insight into the process of convergence.[2] The confidence intervals represent the 30th and 70th percentile run for the respective iteration count. Further experimental details and results can be found in the appendix. The main takeaways are the following:

- Gurobi fails to converge beyond small instances ($\leq$100k nodes for simulation, and $\leq$20k otherwise). Moreover, when it converges, the time required to terminate is multiple orders of magnitude more than that of the first-order optimizers. This is despite the fact that Gurobi is based on an optimized C++ implementation whereas our first-order optimizers implementations are in Python.
- Interestingly, in all such cases in Table 1, where we know the optimal value, the first-order optimizers converge to an optimal strategy. As expected, we can also find some experiments where this is not the case (*e.g.* Rand-42k once Gurobi would eventually terminate). Indeed, we construct an example in the appendix for which our gradient descent and regret matching algorithms all converge to a KKT point that is arbitrarily bad in value relative to the global optimum.
- The RM family of algorithms, and RM$^+$ in particular, consistently outperform GD and OGD in runtime. The difference is often many orders of magnitude, especially in the larger instances.
- RM$^+$ performs best among the RM family. Surprisingly, it typically outperforms PRM$^+$; this stands in stark contrast to what has been observed in zero-sum games [Farina et al., 2021].
- RM$^+$ oftentimes attains higher values than GD and OGD, and almost never less.

6 FUTURE RESEARCH

Our paper opens many interesting avenues for future work. First, we have focused exclusively on solving tabular imperfect-recall decision problems. A promising direction is to use modern RL techniques to expand the scope to even larger problems that cannot be represented in tabular form. Considering other formulations beyond tree-form decision problems, such as (PO)MDPs, is another natural direction that was beyond our scope. Finally, our experiments revealed that the regret matching family of algorithms is a formidable

[1] With the only exceptions of Det-{9m,10m,18m} problems, which we run for 12 hours since the standard time limit poses a significant bottleneck for those instances.

[2] Our regret matching implementations complete more iterations per time than our gradient descent implementations, so the fact that we plot against iterations rather than time favors the gradient descent algorithms.

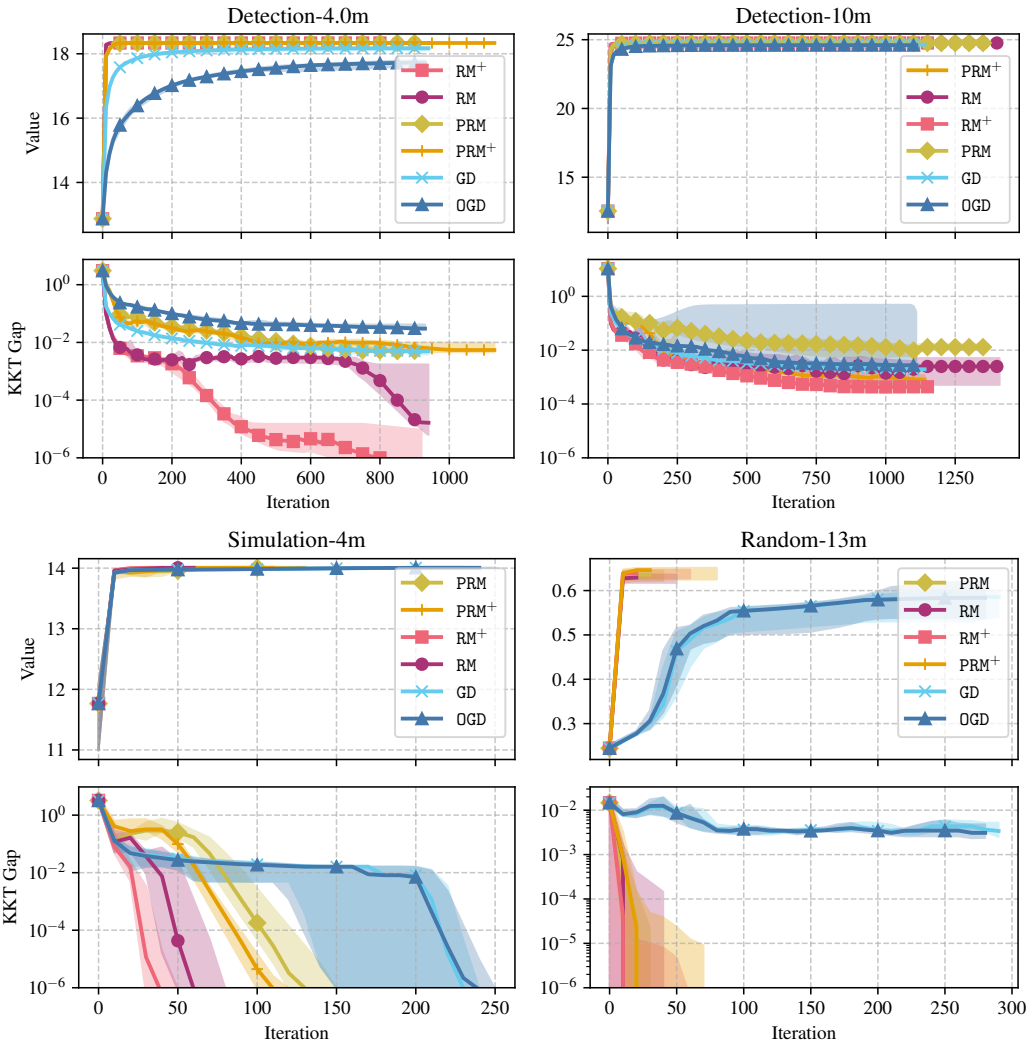

Figure 4: Two detection instances, a simulation and a random instance. They have ~1200 infosets, ~300 infosets, 3 infosets, and ~100 infosets respectively.

first-order optimizer; elucidating their theoretical properties is another important open question.

Acknowledgements

We are grateful to the anonymous reviewers for their valuable improvement suggestions for this paper. Emanuel Tewolde and Vincent Conitzer thank the Cooperative AI Foundation, Macroscopic Ventures and Jaan Tallinn's donor-advised fund at Founders Pledge for financial support. Emanuel Tewolde is also supported in part by the Cooperative AI PhD Fellowship. Tuomas Sandholm and his students Ioannis Anagnostides and Brian Hu Zhang are supported by the Vannevar Bush Faculty Fellowship ONR N00014-23-1-2876, National Science Foundation grants RI-2312342 and RI-1901403, ARO award W911NF2210266, and NIH award A240108S001. Brian Hu Zhang is also supported in part by the CMU Computer Science Department Hans Berliner PhD Student Fellowship.

References

Kenneth Arrow. Economic welfare and the allocation of resources for invention. In *The Rate and Direction of Inventive Activity: Economic and Social Factors*, pages 609–626. National Bureau of Economic Research, Inc, 1962.

Nicola Basilico, Andrea Celli, Giuseppe De Nittis, and Nicola Gatti. Team-maxmin equilibrium: Efficiency bounds and algorithms. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

Heymann Benjamin and Marc Lanctot. Learning in games with progressive hiding. *arXiv:2409.03875*, 2024.

Ratip Emin Berker, Emanuel Tewolde, Ioannis Anagnostides, Tuomas Sandholm, and Vincent Conitzer. The value of recall in extensive-form games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

David Blackwell. An analog of the minmax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6:1–8, 1956.

Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, 2015.

Rachael Briggs. Putting a value on beauty. In Tamar Szabó Gendler and John Hawthorne, editors, *Oxford Studies in Epistemology: Volume 3*, pages 3–34. Oxford University Press, 2010.

Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

Noam Brown and Tuomas Sandholm. Superhuman AI for multiplayer poker. *Science*, 365(6456):885–890, 2019.

Eric O. Chen, Alexis Ghersengorin, and Sami Petersen. Imperfect recall and AI delegation, 2024.

Emery Cooper, Caspar Oesterheld, and Vincent Conitzer. Characterising simulation-based program equilibria. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

Constantinos Daskalakis and Ioannis Panageas. The limit points of (optimistic) gradient descent in min-max optimization. In *NeurIPS 2018*, pages 9256–9266, 2018.

Constantinos Daskalakis and Christos Papadimitriou. Continuous local search. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2011.

Constantinos Daskalakis, Andrew Ilyas, Vasilis Syrgkanis, and Haoyang Zeng. Training gans with optimism. In *International Conference on Learning Representations (ICLR)*, 2018.

Adam Elga. Self-locating belief and the Sleeping Beauty problem. *Analysis*, 60(2):143–147, 2000.

Scott Emmons, Caspar Oesterheld, Andrew Critch, Vincent Conitzer, and Stuart Russell. For learning in symmetric teams, local optima are global nash equilibria. In *International Conference on Machine Learning (ICML)*, 2022.

European Parliament and Council of the EU. Regulation (EU) 2016/679 (General Data Protection Regulation), 2016. URL `https://eur-lex.europa.eu/eli/reg/2016/679/oj`. Official Journal of the European Union.

Gabriele Farina, Andrea Celli, Nicola Gatti, and Tuomas Sandholm. Ex ante coordination and collusion in zero-sum multi-player extensive-form games. In *Neural Information Processing Systems (NeurIPS)*, 2018.

Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Faster game solving via predictive Blackwell approachability: Connecting regret matching and mirror descent. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Gabriele Farina, Julien Grand-Clément, Christian Kroer, Chung-Wei Lee, and Haipeng Luo. Regret matching+: (in)stability and fast convergence in games. In *Neural Information Processing Systems (NeurIPS)*, 2023.

John Fearnley, Paul Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: CLS = PPAD ∩ PLS. *Journal of the ACM*, 70(1):7:1–7:74, 2023.

Dean P. Foster and Sergiu Hart. Smooth calibration, leaky forecasts, finite recall, and nash dynamics. *Games and Economic Behavior*, 109:271–293, 2018.

Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, 1991.

Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68:1127–1150, 2000.

Sergiu Hart and Andreu Mas-Colell. Regret-based continuous-time dynamics. *Games and Economic Behavior*, 45(2):375–394, 2003.

J. R. Isbell. *Finitary Games*, pages 79–96. Princeton University Press, 1957.

Megan Kinniment, Lucas Jun Koba Sato, Haoxing Du, Brian Goodrich, Max Hasin, Lawrence Chan, Luke Harold Miles, Tao R. Lin, Hjalmar Wijk, Joel Burget, Aaron Ho, Elizabeth Barnes, and Paul Christiano. Evaluating language-model agents on realistic autonomous tasks. *arXiv:2312.11671*, 2024.

Daphne Koller and Nimrod Megiddo. The complexity of two-person zero-sum games in extensive form. *Games and Economic Behavior*, 4(4):528–552, 1992.

Vojtěch Kovařík, Caspar Oesterheld, and Vincent Conitzer. Game theory with simulation of other players. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

Vojtěch Kovařík, Caspar Oesterheld, and Vincent Conitzer. Recursive joint simulation in games. *arXiv:2402.08128*, 2024.

Vojtěch Kovařík, Nathaniel Sauerberg, Lewis Hammond, and Ṽincent Conitzer. Game theory with simulation in the presence of unpredictable randomisation. In *Autonomous Agents and Multi-Agent Systems*, 2025.

Christian Kroer and Tuomas Sandholm. Extensive-form game abstraction with bounds. In *ACM Conference on Economics and Computation (EC)*, 2014.

Christian Kroer and Tuomas Sandholm. Imperfect-recall abstractions with bounds in games. In *ACM Conference on Economics and Computation (EC)*, 2016.

H. W. Kuhn. Extensive games and the problem of information. In *Contributions to the Theory of Games*, volume 2 of *Annals of Mathematics Studies, 28*, pages 193–216. Princeton University Press, 1953.

Nicolas S Lambert, Adrian Marple, and Yoav Shoham. On equilibria in games with imperfect recall. *Games and Economic Behavior*, 113:164–185, 2019.

Marc Lanctot, Richard Gibson, Neil Burch, Martin Zinkevich, and Michael Bowling. No-regret learning in extensive-form games with imperfect recall. In *International Conference on Machine Learning (ICML)*, 2012.

Mingyang Liu, Gabriele Farina, and Asuman Ozdaglar. LiteEFG: An efficient python library for solving extensive-form games. *arXiv:2407.20351*, 2024.

Tai-Yu Ma and Philippe Gerber. Distributed regret matching algorithm for dynamic congestion games with information provision. *Transportation Research Procedia*, 3: 3–12, 2014.

Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

Richard R Nelson. The simple economics of basic scientific research. *Journal of political economy*, 67(3):297–306, 1959.

Caspar Oesterheld. Robust program equilibrium. *Theory and Decision*, 86(1):143–159, 2 2019.

Caspar Oesterheld and Vincent Conitzer. Can *de se* choice be *ex ante* reasonable in games of imperfect recall? a complete analysis. `https://www.andrew.cmu.edu/user/coesterh/DeSeVsExAnte.pdf`, 2024. Working paper. Accessed: 2024-07-13.

Alexander Pan, Jun Shern Chan, Andy Zou, Nathaniel Li, Steven Basart, Thomas Woodside, Hanlin Zhang, Scott Emmons, and Dan Hendrycks. Do the rewards justify the means? measuring trade-offs between rewards and ethical behavior in the machiavelli benchmark. In *International Conference on Machine Learning (ICML)*, 2023.

Christos H. Papadimitriou and Mihalis Yannakakis. On complexity as bounded rationality (extended abstract). In *Symposium on Theory of Computing (STOC)*, 1994.

Michele Piccione and Ariel Rubinstein. On the interpretation of decision problems with imperfect recall. *Games and Economic Behavior*, 20:3–24, 1997.

L.D. Popov. A modification to the Arrow-Hurwicz method for search of saddle-points. *Mathematical Notes of the Academy of Sciences of the USSR*, 28(5):845–848, 1980.

Ariel Rubinstein. *Modeling Bounded Rationality*. MIT Press, 1998.

Iosif Sakos, Stefanos Leonardos, Stelios Andrew Stavroulakis, Will Overman, Ioannis Panageas, and Georgios Piliouras. Beating price of anarchy and gradient descent without regret in potential games. In *International Conference on Learning Representations (ICLR)*, 2024.

Matthew Stephenson, Andrew Miller, Xyn Sun, Bhargav Annem, and Rohan Parikh. NDAI agreements. *arXiv:2502.07924*, 2025.

Moshe Tennenholtz. Program equilibrium. *Games and Economic Behavior*, 49(2):363–373, 2004.

Emanuel Tewolde, Caspar Oesterheld, Vincent Conitzer, and Paul W. Goldberg. The computational complexity of single-player imperfect-recall games. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

Emanuel Tewolde, Brian Hu Zhang, Caspar Oesterheld, Manolis Zampetakis, Tuomas Sandholm, Paul W. Goldberg, and Vincent Conitzer. Imperfect-recall games: Equilibrium concepts and their complexity. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2024.

Emanuel Tewolde, Brian Hu Zhang, Caspar Oesterheld, Tuomas Sandholm, and Vincent Conitzer. Computing game symmetries and equilibria that respect them. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

Bernhard Von Stengel and Daphne Koller. Team-maxmin equilibria. *Games and Economic Behavior*, 21(1-2):309–321, 1997.

Kevin Waugh. Abstraction in large extensive games. Master's thesis, University of Alberta, 2009.

Kevin Waugh, Martin Zinkevich, Michael Johanson, Morgan Kan, David Schnizlein, and Michael Bowling. A practical use of imperfect recall. In *Symposium on Abstraction, Reformulation and Approximation (SARA)*, 2009.

Chen-Yu Wei, Chung-Wei Lee, Mengxiao Zhang, and Haipeng Luo. Linear last-iterate convergence in constrained saddle-point optimization. In *International Conference on Learning Representations (ICLR)*, 2021.

Brian Hu Zhang and Tuomas Sandholm. Polynomial-time optimal equilibria with a mediator in extensive-form games. In *Neural Information Processing Systems (NeurIPS)*, 2022.

Brian Hu Zhang, Gabriele Farina, and Tuomas Sandholm. Team belief DAG: generalizing the sequence form to team games for fast computation of correlated team max-min equilibria via regret minimization. In *International Conference on Machine Learning (ICML)*, 2023.

Youzhi Zhang, Bo An, and V. S. Subrahmanian. Correlation-based algorithm for team-maxmin equilibrium in multiplayer extensive-form games. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.

Martin Zinkevich, Michael Bowling, Michael Johanson, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Neural Information Processing Systems (NIPS)*, 2007.

A APPENDIX

Here, we expand further on details we omitted in the main body.

A.1 FURTHER RELATED WORK

Simulation games Commencing from the paper of Kovařík et al. [2023], there has been significant interest in situations where one player can simulate another player [Chen et al., 2024, Kovařík et al., 2024, 2025, Oesterheld, 2019, Cooper et al., 2025]; this is precisely the type of problem captured by one of our benchmarks. The premise of simulating the other player is strongly connected with the notion of *program equilibrium* [Tennenholtz, 2004], where players are allowed to submit source code. This turns out to unlock more cooperative outcomes by expanding the set of equilibria.

MDPs and repeated games Another notable motivation for examining imperfect-recall decision problems lies in the fact that they can result in simpler and more interpretable strategies. This point can be illustrated well in the context of Markov decision problems (MDPs), where insisting on *Markovian* policies—which depend solely on the state and not the entire history—is particularly common; this can be viewed as an extreme form of imperfect recall. Relatedly, restricting the memory and description complexity of a policy has received a lot of attention in the context of repeated games (*e.g.*, Foster and Hart, 2018, Papadimitriou and Yannakakis, 1994). In certain settings, near-optimal policies are possible even under imperfect recall. More broadly, the question of characterizing the value of recall was recently addressed by Berker et al. [2025].

Regret matching Regret matching and its variants have received a lot of attention in (two-player) zero-sum extensive-form games. In particular, the *counterfactual regret minimization (*`CFR`*)* algorithm, famously introduced by Zinkevich et al. [2007], employs a separate `RM` algorithm for each information set. The CFR framework has spawned a flourishing, and still active, line of work. Yet, much less is known beyond (two-player) zero-sum games. It has to be stressed again that in zero-sum games, `RM` and its variants only have guarantees concerning the time average strategy. In fact, the last iterate can fail to converge [Farina et al., 2023]. Our experiments suggest a fundamental difference in constrained optimization problems: all our results make use of the last iterate, which not only converges, but does so remarkably fast. To our knowledge, there is currently no theory that predicts that `RM` and its variants will converge. The continuous time of `RM` was analyzed by Hart and Mas-Colell [2003], who also established asymptotic convergence in two-player potential games for a certain—somewhat artificial—variant of `RM` in discrete time. Fast empirical convergence was reported by Ma and Gerber [2014] in a certain class of congestion games.

An intriguing behavior we uncover in this paper is that the `RM` family of algorithms often outperforms (`O`)`GD` in terms of the attained value, at least for the benchmark problems we consider. In the context of multi-player potential games, which is closely related to imperfect-recall decision problems, the problem of characterizing the performance of different algorithms is poorly understood. One notable contribution here is the recent paper of Sakos et al. [2024], but it only focused on 2×2 games. Providing a theoretical explanation that justifies the excellent performance of `RM` in terms of value is an interesting but challenging direction for the future.

CDT equilibria The CDT equilibrium falls under the family based on the *multi-selves* approach [Kuhn, 1953]. At a high level, whenever the player has imperfect information on the decision node of an infoset it is currently in, the player will weight each possibility with the probability of reaching the decision node in question under strategy $\boldsymbol{x}$. The name is derived from the intuition that the player's choice to deviate from $\boldsymbol{x}$ at the current node does not *cause* any change in its behavior at any other node, even if they are of the same infoset. Another prominent member is the *EDT equilibrium*, which results from marrying evidential decision theory with the multi-selves approach [Oesterheld and Conitzer, 2024]. For further background on the ongoing debate around decision theories and how they relate to belief formation (*cf.* the "sleeping beauty" problem [Elga, 2000]), we refer to [Piccione and Rubinstein, 1997, Briggs, 2010, Oesterheld and Conitzer, 2024]. Further, we refer to Tewolde et al. [2024] for a computational treatment of equilibria in multi-player games with imperfect recall. With regard to the complexity of computing CDT equilibria, we saw earlier that a $\mathsf{poly}(1/\epsilon)$ time algorithm exists by running `GD` on a suitable optimization problem; in the regime where ϵ is exponentially small, the complexity is characterized by the class `CLS`, and is believed to be hard [Daskalakis and Papadimitriou, 2011, Fearnley et al., 2023, Tewolde et al., 2023]. Conceptually, and also computationally, CDT equilibria in decision problems with imperfect recall have been also connected to Nash equilibria in team games that respect a given set of game symmetries [Lambert et al., 2019, Emmons et al., 2022, Tewolde et al., 2025].

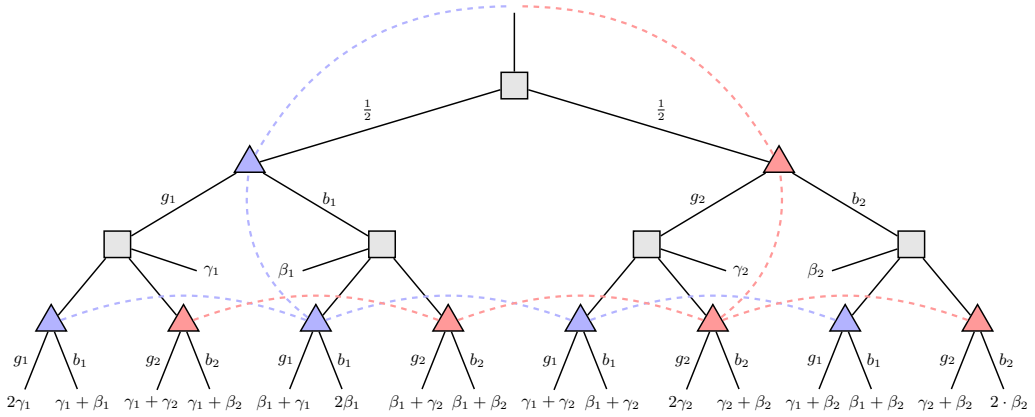

Figure 5: Deployment phase Γ' of the more complex simulation problem with two scenarios given in Figure 2 (bottom). In deployment, the agent acts at least once and up to two times in total. The "good" and "bad" actions yield different immediate payoffs in different scenarios, and they contribute additively to the total payoffs at terminal nodes.

Mixed strategies and team games Much of the prior work in extensive-form games has focused on *mixed* strategies—probability distributions over pure strategies. Unlike behavioral strategies, mixed strategies allow the player to correlate its actions across infosets; one such example is *ex-ante* team coordination [Farina et al., 2018] in the context of team games. As we explained in our introduction, a team game can be phrased as an imperfect-recall decision problem; in fact, one without absentmindedness. Without absentmindedness, it follows that there exists an optimal strategy that is pure; in contrast, the presence of absentmindedness—which is primary focus on this paper—requires randomization [Isbell, 1957]. In the presence of imperfect recall, mixed strategies are not realization-equivalent to behavioral strategies [Kuhn, 1953], and they do not fit our motivation since they imply a form of memory mechanism. Related to *ex-ante* team coordination, classical equilibrium concepts in extensive-form games involving *correlation* can be modeled via a mediator—a trusted third party—with imperfect recall [Zhang and Sandholm, 2022]; that the mediator has imperfect recall can serve to safeguard the players' sensitive information, which is tied to one of the key motivations of this paper.

A.2 DEPLOYMENT PHASE OF SIMULATION PROBLEMS

Figure 5 displays the subgame Γ' representing the deployment phase of the simulation problem we start to describe in Figure 2 (bottom).

A.3 WHEN FIRST-ORDER OPTIMIZERS PERFORM POORLY

We find it quite surprising that the first-order optimizers we benchmark perform so well in terms of utility value in comparison to the global optimum found by `Gurobi`. Indeed, from a theoretical standpoint, we can give examples in which the `RM` and `GD` families of algorithms converge to an arbitrarily bad value relative to the global optimum. To illustrate, consider the (ϵ, k)-parametrized function $f_{\epsilon,k}(x) = \frac{1}{\epsilon}(x^k - \epsilon)^2$ for $\epsilon > 0$ and $k \in \mathbb{N}$. We can investigate maximizing this polynomial function over the unit interval. With some slight adjustments, this will be equivalent to a polynomial optimization problem over the 1-simplex which, in return, is equivalent to a decision problem with imperfect recall [Tewolde et al., 2023]. The function f is plotted in Figure 6 for $\epsilon = 0.1$ and multiple values for k. In all cases, $f_{\epsilon,k}(x) \geq 0$, $f_{\epsilon,k}(0) = \epsilon$, and $f_{\epsilon,k}(1) > 1$ if we additionally restrict $\epsilon < \frac{3-\sqrt{5}}{2} \approx 0.382$. If an algorithm therefore converges to $x^* = 0$, we have found a family of instances for which the algorithm has achieved no more than MIN $+ \epsilon \cdot$ (MAX $-$ MIN) in value, where MAX and MIN represent the max and min values f on $[0, 1]$ (or, respectively, the utility function on the 1-simplex). For our first-order methods, note that $f_{\epsilon,k}$ is strictly decreasing in the interval $J = [0, \epsilon^{1/k})$. If the `RM` and `GD` families of algorithms are therefore initialized to start in J, they will converge to 0. Assuming we draw the initial point uniformly random from $[0, 1]$, this situation occurs with probability $\epsilon^{1/k}$. Therefore, we can first set the desired poor-performance parameter ϵ, and then $k = k(\epsilon)$ to meet the desired probability confidence $\epsilon^{1/k}$, to obtain arbitrarily bad performance of the `RM` and `GD` families of algorithm with arbitrarily high probability in some instance.

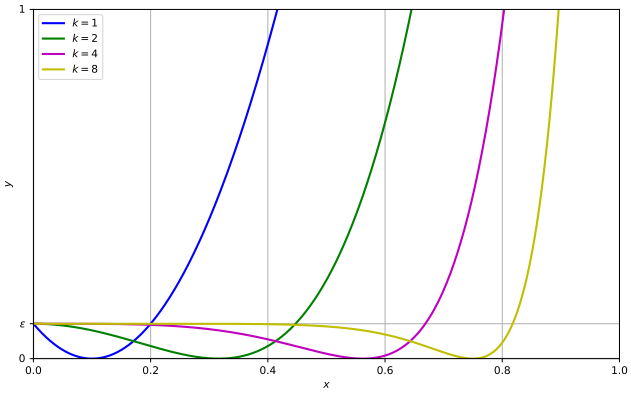

Figure 6: A family of polynomial optimization problems over the unit interval on which the RM and GD families of algorithms perform arbitrarily poorly.

A.4 ADDITIONAL EXPERIMENTAL DETAILS AND RESULTS

All experiments were run on a 64-core AMD Opteron 6272 processor. Each run was allocated one thread with a maximum of 16GBs of RAM. The commercial solver Gurobi requires a license to run on decision problems of nontrivial size. The result table of the experiments for the full set of benchmark decision problems is given in Table 2, which now also includes PRM experiments. We display "—" in the time column of Gurobi if it does not converge to the global optimum (up to a tolerance of 10^{-6} within the time limit), and "—" in its value column if it cannot even produce a "best-so-far" strategy within the time limit.[3]

The supplementary code contains the files that can generate decision problems with imperfect recall, solve them with the algorithms we discuss in Section 3, and plot their optimization progress. The particular benchmark instances of Table 2, together with experiments and plots regarding them, are available in the following Google drive link: https://drive.google.com/file/d/1o9uDbbRUa3BOlp-IXYvq68cwfCsWudPr/view?usp=drive_link.

[3]This happens whenever Gurobi spends all of its time on *presolving*, and because we do not supply Gurobi with a strategy initialization.

Problem	Gurobi		GD			OGD			RM			RM+			PRM			PRM+		
	value	time	value	time	gap	value	time	gap	value	time	gap	value	time	gap	value	time	gap	value	time	gap
Det-86	18.00	0.22s	18.00	0.01s	—	18.00	0.01s	—	18.00	0.00s	—	18.00	0.00s	—	18.00	0.00s	—	18.00	0.00s	—
Det-105	12.00	1.86s	12.00	0.01s	—	12.00	0.01s	—	12.00	0.00s	—	12.00	0.00s	—	12.00	0.00s	—	12.00	0.00s	—
Det-1k	13.00	1m 24s	13.00	0.13s	—	13.00	0.07s	—	13.00	0.32s	—	13.00	0.36s	—	13.00	0.38s	—	13.00	0.41s	—
Det-1.8k	22.00	2m 40s	22.00	0.06s	—	22.00	0.07s	—	22.00	0.03s	—	22.00	0.03s	—	22.00	0.03s	—	22.00	0.03s	—
Det-2.0k	17.50	1m 42s	17.50	0.03s	—	17.50	0.05s	—	17.50	0.03s	—	17.50	0.03s	—	17.50	0.03s	—	17.50	0.03s	—
Det-8k	16.67	—	16.62	13.05s	—	16.62	2m 36s	—	16.67	—	3e-05	16.67	2m 39s	—	16.67	—	0.001	16.67	—	0.007
Det-10.6k	12.84	—	12.70	24.87s	—	12.70	5m 0s	—	12.84	6.41s	—	12.84	6.74s	—	12.84	15.53s	—	12.84	14.48s	—
Det-10.7k	20.20	16m 36s	20.20	0.21s	—	20.20	0.23s	—	20.20	0.73s	—	20.20	0.77s	—	20.20	1.06s	—	20.20	1.13s	—
Det-86k	14.89	—	14.84	—	0.004	10.00	—	11.4	14.89	2m 7s	—	14.89	2m 4s	—	14.89	7m 54s	—	14.89	5m 50s	—
Det-130k	15.53	—	15.37	—	8e-06	15.40	45m 59s	—	15.53	5m 38s	—	15.53	2m 3s	—	15.53	—	0.0001	15.53	—	8e-05
Det-139k	18.89	—	18.76	15m 24s	—	18.76	16m 48s	—	18.89	1m 47s	—	18.89	1m 50s	—	18.89	3m 29s	—	18.89	3m 10s	—
Det-718k	—	—	12.76	—	0.0005	12.69	—	0.005	12.84	30m 55s	—	12.84	31m 21s	—	12.84	—	0.001	12.84	—	0.0008
Det-1.002m	—	—	13.93	—	0.0003	13.90	—	0.005	13.96	15m 36s	—	13.96	17m 17s	—	13.96	40m 10s	—	13.96	35m 35s	—
Det-1.008m	—	—	12.64	—	0.0006	12.54	—	0.008	12.75	33m 31s	—	12.75	22m 38s	—	12.75	54m 51s	—	12.75	31m 28s	—
Det-2.1m	—	—	26.00	—	1e-05	25.96	—	0.02	26.15	—	0.003	26.15	3h 25m	—	26.15	—	0.006	26.15	—	0.005
Det-2.2m	—	—	16.20	—	0.002	15.93	—	0.02	16.36	2h 22m	—	16.36	3h 13m	—	16.36	—	2e-06	16.36	—	5e-06
Det-3.8m	—	—	15.66	—	0.003	15.14	—	0.03	15.80	—	2e-06	15.80	—	5e-05	15.80	—	0.002	15.80	—	0.0003
Det-4.0m	—	—	18.17	—	0.005	17.72	—	0.03	18.34	—	2e-05	18.34	2h 55m	—	18.34	—	0.005	18.34	—	0.005
Det-4.1m	—	—	17.88	—	0.003	17.47	—	0.03	18.06	—	4e-05	18.06	—	2e-05	18.06	—	0.003	18.06	—	0.0007
Det-4.2m	—	—	19.98	—	0.003	20.07	—	0.003	20.15	—	0.0004	20.15	—	2e-05	20.15	—	0.01	20.15	—	0.02
Det-9m	—	—	23.16	—	0.004	22.71	—	0.02	23.45	—	0.0001	23.45	—	0.0001	23.45	—	0.0003	23.45	—	0.0004
Det-10m	—	—	24.64	—	0.002	24.61	—	0.003	24.76	—	0.002	24.76	—	0.0004	24.76	—	0.01	24.76	—	0.0008
Det-18m	—	—	26.38	—	0.006	25.81	—	0.05	26.71	—	0.004	26.71	—	0.001	26.71	—	0.04	26.71	—	0.04
Rand-7k	0.53	25m 18s	0.49	4.88s	—	0.49	5.14s	—	0.50	0.38s	—	0.50	0.27s	—	0.50	0.26s	—	0.50	0.34s	—
Rand-11.9k	1.00	1h 16m	0.97	0.93s	—	0.97	0.90s	—	0.95	0.26s	—	0.95	0.29s	—	0.95	0.19s	—	0.95	0.23s	—
Rand-12.2k	1.00	1h 52m	0.93	3.33s	—	0.92	2.73s	—	0.93	0.36s	—	0.93	0.41s	—	0.94	0.36s	—	0.94	0.41s	—
Rand-24k	0.72	—	0.66	7m 0s	—	0.66	7m 46s	—	0.66	26.55s	—	0.66	1m 3s	—	0.66	1m 54s	—	0.66	5m 5s	—
Rand-35k	1.00	—	0.95	3.85s	—	0.95	3.76s	—	0.92	0.99s	—	0.92	1.18s	—	0.92	0.92s	—	0.94	1.68s	—
Rand-42k	0.69	—	0.55	—	0.01	0.55	—	0.01	0.65	—	2e-06	0.65	5m 56s	—	0.65	—	5e-06	0.65	3m 19s	—
Rand-165k	0.37	—	0.96	19.77s	—	0.97	18.48s	—	0.96	4.33s	—	0.97	4.95s	—	0.96	5.24s	—	0.90	4.02s	—
Rand-179k	0.38	—	0.88	—	0.0003	0.88	—	1e-06	0.94	5.97s	—	0.93	10.27s	—	0.93	6.66s	—	0.91	7.31s	—
Rand-198k	0.40	—	0.96	25.37s	—	0.95	22.61s	—	0.96	8.10s	—	0.96	7.41s	—	0.95	5.22s	—	0.96	6.31s	—
Rand-1.2m	—	—	0.93	2m 46s	—	0.93	2m 28s	—	0.96	35.86s	—	0.97	36.07s	—	0.96	31.74s	—	0.96	31.09s	—
Rand-1.3m	—	—	0.96	4m 0s	—	0.96	3m 17s	—	0.96	2m 26s	—	0.96	54.77s	—	0.98	1m 33s	—	0.93	38.99s	—
Rand-2m	—	—	0.92	3m 53s	—	0.93	3m 44s	—	0.94	59.29s	—	0.93	1m 21s	—	0.96	2m 1s	—	0.96	58.84s	—
Rand-4m	—	—	0.94	13m 1s	—	0.94	15m 39s	—	0.93	3m 12s	—	0.92	4m 26s	—	0.92	8m 41s	—	0.93	4m 16s	—
Rand-6m	—	—	0.97	17m 34s	—	0.97	15m 40s	—	0.98	2m 30s	—	0.98	2m 9s	—	0.98	2m 50s	—	0.98	2m 10s	—
Rand-7m	—	—	0.97	22m 27s	—	0.98	25m 7s	—	0.94	2m 13s	—	0.93	2m 52s	—	0.96	2m 55s	—	0.97	3m 47s	—
Rand-13m	—	—	0.59	—	0.003	0.58	—	0.003	0.63	19m 11s	—	0.64	17m 31s	—	0.64	20m 39s	—	0.65	36m 42s	—
Rand-18m	—	—	0.97	2h 33m	—	0.97	3h 0m	—	0.95	29m 45s	—	0.97	24m 0s	—	0.96	13m 8s	—	0.97	14m 31s	—
Rand-23m	—	—	0.94	3h 37m	—	0.93	—	0.0007	0.98	23m 10s	—	0.96	23m 5s	—	0.98	16m 48s	—	0.95	18m 2s	—
Sim-245	4.41	0.18s	4.41	0.00s	—	4.41	0.00s	—	4.41	0.00s	—	4.41	0.00s	—	4.41	0.00s	—	4.41	0.00s	—
Sim-438	7.21	0.41s	7.21	0.00s	—	7.21	0.00s	—	7.21	0.00s	—	7.21	0.00s	—	7.21	0.00s	—	7.21	0.00s	—
Sim-759	3.89	2.97s	3.89	0.01s	—	3.89	0.01s	—	3.89	0.01s	—	3.89	0.01s	—	3.89	0.01s	—	3.89	0.01s	—
Sim-3k	6.25	1m 1s	6.25	0.32s	—	6.25	1.03s	—	6.25	0.26s	—	6.25	0.28s	—	6.25	0.52s	—	6.25	0.48s	—
Sim-7k	8.58	1m 36s	8.58	0.05s	—	8.58	0.05s	—	8.58	0.05s	—	8.58	0.05s	—	8.58	0.05s	—	8.58	0.05s	—
Sim-13k	10.38	4m 21s	10.38	0.69s	—	10.38	8.54s	—	10.38	1.03s	—	10.38	1.01s	—	10.38	4.75s	—	10.38	3.97s	—
Sim-34k	10.44	1h 42m	10.44	4.89s	—	10.44	6.74s	—	10.44	2.52s	—	10.44	2.81s	—	10.44	5.03s	—	10.44	5.01s	—
Sim-66k	6.94	1h 31m	6.94	5.63s	—	6.94	8.70s	—	6.94	5.51s	—	6.94	3.94s	—	6.94	17.32s	—	6.94	15.01s	—
Sim-105k	4.40	—	4.40	18.60s	—	4.40	1m 0s	—	4.40	2m 41s	—	4.40	55.90s	—	4.40	15m 18s	—	4.40	10m 51s	—
Sim-125k	14.47	—	14.48	12.70s	—	14.48	19.76s	—	14.48	11.70s	—	14.48	12.15s	—	14.48	18.83s	—	14.48	19.68s	—
Sim-226k	8.57	—	9.70	2.16s	—	9.70	4.28s	—	9.70	1.52s	—	9.70	1.50s	—	9.70	1.49s	—	9.70	1.48s	—
Sim-415k	6.30	—	8.81	3.85s	—	8.81	3.43s	—	8.81	2.65s	—	8.81	2.65s	—	8.81	2.65s	—	8.81	2.64s	—
Sim-441k	11.79	—	13.57	57.23s	—	13.57	1m 15s	—	13.57	36.88s	—	13.57	33.94s	—	13.57	2m 38s	—	13.57	1m 35s	—
Sim-540k	6.41	—	8.54	47.54s	—	8.54	2m 37s	—	8.54	19.39s	—	8.54	19.44s	—	8.54	3m 48s	—	8.54	3m 3s	—
Sim-866k	8.77	—	10.49	2m 4s	—	10.49	2m 27s	—	10.49	1m 31s	—	10.49	1m 0s	—	10.49	2h 34m	—	10.49	2h 0m	—
Sim-1m	4.14	—	4.77	5m 33s	—	4.77	7m 2s	—	4.77	2m 14s	—	4.77	2m 34s	—	4.77	4m 16s	—	4.77	4m 20s	—
Sim-1.7m	11.05	—	13.33	10m 26s	—	13.33	11m 16s	—	13.33	4m 28s	—	13.33	4m 53s	—	13.33	7m 22s	—	13.33	7m 12s	—
Sim-1.9m	—	—	13.45	18.31s	—	13.45	17.96s	—	13.45	12.36s	—	13.45	12.19s	—	13.45	12.48s	—	13.45	12.47s	—
Sim-2.3m	—	—	11.09	22.01s	—	11.09	21.88s	—	11.09	14.97s	—	11.09	15.00s	—	11.09	15.01s	—	11.09	15.13s	—
Sim-4m	—	—	14.01	45m 5s	—	14.01	41m 0s	—	14.01	11m 36s	—	14.01	7m 3s	—	14.01	26m 45s	—	14.01	21m 17s	—

Table 2: Experimental results for the full set of benchmarks.