Evolutionary Multi-Armed Bandits with Genetic Thompson Sampling

Baihan Lin

▶ **To cite this version:**

Baihan Lin. Evolutionary Multi-Armed Bandits with Genetic Thompson Sampling. 2022 IEEE Congress on Evolutionary Computation (CEC), Jul 2022, Padova, Italy. ⟨hal-03762294⟩

HAL Id: hal-03762294

https://hal.science/hal-03762294v1

Submitted on 27 Aug 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

Evolutionary Multi-Armed Bandits with Genetic Thompson Sampling

Baihan Lin
Columbia University
baihan.lin@columbia.edu

arXiv:2205.10113v1 [cs.NE] 26 Apr 2022

***Abstract*—As two popular schools of machine learning, online learning and evolutionary computations have become two important driving forces behind real-world decision making engines for applications in biomedicine, economics, and engineering fields. Although there are prior work that utilizes bandits to improve evolutionary algorithms' optimization process, it remains a field of blank on how evolutionary approach can help improve the sequential decision making tasks of online learning agents such as the multi-armed bandits. In this work, we propose the Genetic Thompson Sampling, a bandit algorithm that keeps a population of agents and update them with genetic principles such as elite selection, crossover and mutations. Empirical results in multi-armed bandit simulation environments and a practical epidemic control problem suggest that by incorporating the genetic algorithm into the bandit algorithm, our method significantly outperforms the baselines in nonstationary settings. Lastly, we introduce EvoBandit, a web-based interactive visualization to guide the readers through the entire learning process and perform lightweight evaluations on the fly. We hope to engage researchers into this growing field of research with this investigation.**

***Index Terms*—Multi-armed bandits, genetic algorithm**

I. Introduction

As an important practical problem in many real-world applications, online learning solves the challenge that the data is only revealed in a sequential fashion and subsequently used to update the best predictive model for unseen future reward or data corresponding to the features of the data. If we consider the many real-world scenarios of this, the reward feedback serves as the only place for the agent to efficiently learn from the available historical experience in a sequential order. The sequential decision making is a field where this setting is especially important. It concerns with an environment where the agent needs to select the best available action to take at each iteration in order to maximize the cumulative reward across a period of time. The key to the solution of this problem is to find an optimal trade-off between two processes in the decision making: the exploitation of the learned reward correspondence from the known actions and the exploration of unfamiliar actions. The *Multi-Armed Bandits (MAB)* problem describes the mathematical formulation of this framework where each bandit arm maps to a usually fixed but always unknown reward distribution [1], [2], and at each step the player picks an arm to play, gets a reward feedback and updates itself accord to the feedback. These online learning agents update from trials and errors based on feedback in a temporal sequence, and are widely applied to applications such as user modeling, recommendation systems, epidemic control and speaker diarization [3]–[9].

Evolutionary computation, on the other hand, solves problem with population-based trials and errors. Usually inspired by biological evolution, these global optimization learners usually start with a pool of candidate solutions and iteratively update them by stochastically removing less favorable solutions based on some notion of fitness score and introducing incremental mutations to more favorable ones in a way that mimics the natural selection process [10]. Because these methods collect feedback from a pool of representations instead of merely from a temporal sequence, they usually reach a larger set of solution space, yield multiple optimal solutions, and benefit from the parallelism offered by high-performance distributed computing. Since they often provide highly optimized solutions in realistic scenarios, they are widely applied in engineering [11]–[13].

Despite the popularity of both schools of machine learning research, there haven't been much work describing their intersection, combining the effective learning of sequential data streams based on ongoing feedback and the parallel global optimization endowed by the evolutionary computation. Most of the work in this niche field have been gravitated towards how to improve the optimization criterion of the evolutionary algorithms with bandit approaches such as [14]–[17]. To the best of our knowledge, this is one of the first work that combines the evolutionary computational components into the online learning problem in order to directly improve the bandit algorithms. We build upon the Thompson Sampling [18] and propose a genetic algorithm variant, called the Genetic Thompson Sampling (GTS). We evaluate the algorithm in a series of simulation environment and demonstrate a clear advantage over existing bandit algorithms in nonstationary multi-armed bandits scenarios. Lastly, we present EvoBandit, a web-based visualization app where the users can navigate the learning process of the agent through interactive visual storytelling, play around different hyperparameters in this lightweight simulation environment, and hopefully get interested in this growing field.

II. Problem Setting

The *Multi-Armed Bandit (MAB)* problem describes a sequential decision making process with reinforcement learning, where at each step the agent selects an action from a finite action set and aims to maximize the cumulative reward over time (Algorithm 1). There have been multiple optimal solutions proposed in either stochastic [1], [2] or adversarial [19]–[21] formulations. Among the formulations, the Bayesian formulation attracts attention lately [22] with an algorithm

Algorithm 1 The Multi-Armed Bandit Problem

for t = 1,2,3,$\cdots$, T **do**
  $r(t)$ is drawn according to $\mathbb{P}_r$
  Player chooses an action $a = \pi_t(t)$
  Feedback $r_a(t)$ for only the chosen arm is revealed.
  Player updates its policy π_t
end for

Algorithm 2 Thompson Sampling (TS)

Initialize: $S_{a'} = 1$, $F_{a'} = 1, \forall a' \in A$.
For each episode e **do**
  Initialize state s
  Repeat for each step t of the episode e:
    Sample $\theta_{a'} \sim Beta(S_{a'}, F_{a'}), \forall a' \in A_t$
    Take action $a = \arg\max_{a'} \theta_{a'}$, and
    Observe $r \in R_{a'}$
    $S_a := S_a + r$
    $F_a := F_a + (1 - r)$
  until s is the terminal state
End for

called the Thompson sampling [23]. Empirical evaluation [24] and theoretical analysis [25] show that Thompson sampling is asymptotically optimal for Bernoulli bandits and highly competitive for multiple complex problems.

III. Background

A. Thompson Sampling

Thompson Sampling (TS) [18] describes a class of probability matching algorithm within the Bayes-optimal framework of the multi-armed bandits. It attempts to directly maximizes expected cumulative payoffs corresponding to a given prior distribution [22]. As in Algorithm 2, the Thompson Sampling agent picks the arm (i.e. action) that maximizes the expected reward based on a randomly drawn belief, which in this case, is a Beta distribution. For each arm of the bandit, there are two parameters, S which stands for "success" and F which stands for "failure". They are initialized as 1. In each round, the agent select the action with a policy that maximizes a random variable sampled from the Beta distributions of each bandit arm. Then the agent observe a reward feedback ranging from 0 to 1, and use that to update the two parameters S and F by incrementing one, the other, or both, with the reward magnitude. We see that for each arm, the relative difference between the S and F determines the quality of the bandit arm. If the numbers of S and F are small (as in the early rounds of learning), the sample from the Beta distribution is very stochastic, hence promoting the exploration. When there are more evidence accumulated on certain bandit arm, the numbers of S and F will increase to a large values, which make the sample from the Beta distribution more certain.

Algorithm 3 Genetic Algorithm (GA)

Initialize: M_0 as a population of N randomly generated individuals in generation 0. γ as the selection ratio. μ as the mutation rate.
Compute fitness score f_m for $m \in M_0$.
Episode $t = 0$.
Repeat for each step t:
  Create M_t, the population of generation t.
  Selection: Select $\gamma \times N$ members of M_t based on fitness scores and insert into M_{t+1}, the generation $t + 1$.
  Crossover: Create $(1-\gamma) \times N$ new members by pairing M_{t+1} to produce offspring and insert them into M_{t+1}.
  Mutation: Mutate $\mu \times N$ members of M_{t+1}.
  t := t+1
until the fitness of the fittest member in M_t is high enough.

B. Genetic Algorithm

Genetic algorithm (GA) describes a class of evolutionary algorithm that mimics the natural selection process in biology where the genomes within the population evolves by competition, selection, mutation, and crossover of genetic components [26]. As in Algorithm 3, the genetic algorithm maintains an evolving population of candidate solutions. For each generation t, a fitness score is computed for each candidate model $m \in M_t$, and only a subset of the candidate models, deemed as elites fit enough by the fitness scores, are kept in the population of the next generation. The not-so-fit models which doesn't match the criterion are eliminated, making room for new individuals. These fitter models are then used as "parents", to be paired up to generate "offspring". These offspring are introduced into the population of the next generation until it is full. To introduce genetic diversity, the genetic algorithm then performs a number of mutations in randomly selected models in the population. This process is performed generation by generation until the fittest model in the population is fit enough for the problem.

IV. Related work

A. Bandit algorithms and evolutionary computation

There are multiple prior work that utilizes the bandit algorithms to improve the evolutionary computation applications. For instance, N-Tuple Bandit Evolutionary Algorithm [27] describes a hybrid approach that use bandit to balance the exploitation-exploration tradeoff within the search space of a large population of population with many unsampled points. This approach shows merit in game playing [15], automatic game improvement [27], game agent optimizations [16] and modeling player experience [28]. Bandit algorithms have also been applied to adaptively select proper operator on the fly that maximizes the quality measure of the candidate solutions [29]. A similar approach was later applied to more complicate problems such as multi-objective evolutionary computation [30]. On relevant work to ours is [31], where a differential evolution algorithm is proposed to pick an agent among multiple agents whose parameters are unknown. Unlike this black box portfolio selection problem, our method directly optimizes and update the bandit agent with explicit genetic algorithm approaches.

B. Multi-agent bandits

Our usage of multiple bandit agents in a population is related to the literature of multi-agent bandits. Multi-agent networks is a class of bandit algorithms where a pool of agents shares information to one another [32]. Similar to their work, our algorithm also hosts a population of agents, and makes decisions based on the majority of the agent votes. This type of multi-agent network can also be extended to the multi-agent coordination problem, where the agent can perform learning independently from one another in different iterations but still share information to one another. For example, [33] explores the neighborhood structure of such networks to improve the coordination problem. If we consider the agents living in a society, the agents can cooperate with one another in a social learning way as in [34], or decide to cooperate vs. defect in social dilemma situation such as Iterated Prisoners' Dilemma as in [35]. In most cases, the cooperative multi-armed bandits have shown to be beneficial in various settings [36], [37].

C. Mixture of experts models

Having a population of decision making agents is also related to the mixture of experts model [38] which uses multiple expert learning networks to divide a problem space into separate homogeneous regions to conquer individually. The output of such models are usually moderated by multiple levels of probabilistic gating functions [39]. Similar to their approach, among the population of decision making agents, we only adopt the recommendation from a subset of the agents (by taking the majority vote).

D. Routing information among bandit components

Related to the weight sharing among agents in multi-agent networks, bandit algorithms can also be designed to incorporate multiple modules and route information among them during the online learning process. For instance, [40] proposes a contextual bandit algorithm that host multiple embeddings trained online in batches. The agent adaptively select which deep embeddings to use given a specific context and make decisions which in turn changes these embeddings in a routing way. Similarly, in bandits, information can be propagate across the modules when feedback are sparse and therefore unavailable in certain iteration, as in the semi-supervised bandit in [41].

V. Method

In Algorithm 4, we introduce the Genetic Thompson Sampling (GTS). Here are some preliminaries: A is the action space. R is the reward function that assigns rewards from a range of $[0, 1]$. M is a population of Thompson Sampling agents, each with their starting bandit parameters $S^m_{a'}$ and $F^m_{a'}$ for each action $a' \in A$, and their starting fitness parameters S_f and F_f which are both initiated as 1. Q is a real number larger than 1, and the starting bandit parameters $S^m_{a'}$ and $F^m_{a'}$ are assigned by sampling from uniform distribution $U(1, Q)$.

For each step, the pool of Thompson Sampling agents all make their recommendation of which arm to pull. Then the final action selected by the Genetic Thompson Sampling is the majority of these recommendation, i.e. the mode. Then the reward is revealed, only to update those agents whose recommendation align with the action finally chosen by the whole population. The updates have too components, one for the Thompson Sampling bandit agents, one for the fitness parameters S_f and F_f. In another word, if the recommendation of a certain agent is adopted, and it received a positive feedback, the "success" rate of this agent should be higher, and vice versa (the "failure" rate would be higher if this is a negative feedback). Then the fitness score is stochastically computed just like the Thompson Sampling step in the bandit component, that for each agent, a sample from its Beta distribution is created, and then ranked. The rationale is that, the fitness rating can be formulated in a multi-armed bandit problem as a exploration-exploitation tradeoff. We want to select the best candidate solutions for the next step (exploitation), but we also don't want to risk not knowing what is really the best solutions as we might not have enough knowledge yet (exploration). In early round, the fitness measure would likely be less accurate than later round, and thus, should have a smaller S_f and F_f which makes the stochastic sampling more noisy, encouraging more exploration. In later rounds, when we have more knowledge for how fit each candidate model is, the sampling from the Beta distribution becomes more certain.

Algorithm 4 Genetic Thompson Sampling (GTS)

Initialize: M, Q, A, $R \in [0, 1]$.
  For each agent $m \in M$:
    $S^m_f = 1$, $F^m_f = 1$.
    $q \sim U(1, Q), S^m_{a'} = q$, $F^m_{a'} = q, \forall a' \in A$.
  End for
For each episode e **do**
  Initialize state s
  Repeat for each step t of the episode e:
    For each agent $m \in M$:
      Sample $\theta^m_{a'} \sim Beta(S^m_{a'}, F^m_{a'}), \forall a' \in A_t$.
      Get recommendation $a^m = \arg\max_{a'} \theta^m_{a'}$.
    End For
    Take action $a^* = Mo(a)$, and Observe $r \in R_{a^*}$.
    For each agent $m \in \{m \in M | a^m == a^*\}$:
      $S^m_f := S^m_f + r$
      $S^m_f := F^m_f + (1 - r)$
      $S^m_{a^*} := S^m_{a^*} + r$
      $F^m_{a^*} := F^m_{a^*} + (1 - r)$
    End For
    Get fitness score $f^m \sim Beta(S^m_f, F^m_f), \forall m \in M$.
    Selection: $M^p = \text{selection}(M, f)$.
    Crossover: $M^c = \text{crosscover}(M^p)$.
    Mutation: $M = \text{mutation}(M^c)$.
  until s is the terminal state
End for

A. The selection module

There are multiple ways to determine the elites in the population to select for reproduction. In this work, we choose

TABLE I: **MAB evaluation** with increasing population sizes (presented metric is the cumulative reward.)

	Stationary bandits			Nonstationary (NS) bandits		
	MAB-5	MAB-10	MAB-50	NS MAB-5	NS MAB-10	NS MAB-50
Random	53.48 ± 2.58	46.36 ± 1.93	47.76 ± 0.92	66.00 ± 6.77	50.00 ± 7.14	50.00 ± 7.14
TS	$\mathbf{68.40 \pm 2.40}$	67.94 ± 2.55	$\mathbf{63.52 \pm 0.81}$	64.44 ± 2.75	60.20 ± 1.21	54.06 ± 0.92
UCB1	60.62 ± 2.12	55.94 ± 1.97	48.26 ± 0.81	75.98 ± 1.57	60.36 ± 1.01	50.54 ± 0.67
GTS-p10	66.12 ± 2.45	67.64 ± 2.55	62.64 ± 0.74	59.66 ± 1.78	69.26 ± 1.71	58.34 ± 1.38
GTS-p25	67.62 ± 2.45	66.36 ± 2.82	60.80 ± 0.81	59.08 ± 2.76	74.78 ± 2.37	71.06 ± 1.24
GTS-p100	67.64 ± 2.51	$\mathbf{68.00 \pm 2.62}$	61.70 ± 0.83	$\mathbf{89.14 \pm 2.05}$	$\mathbf{93.84 \pm 1.01}$	$\mathbf{89.52 \pm 0.74}$

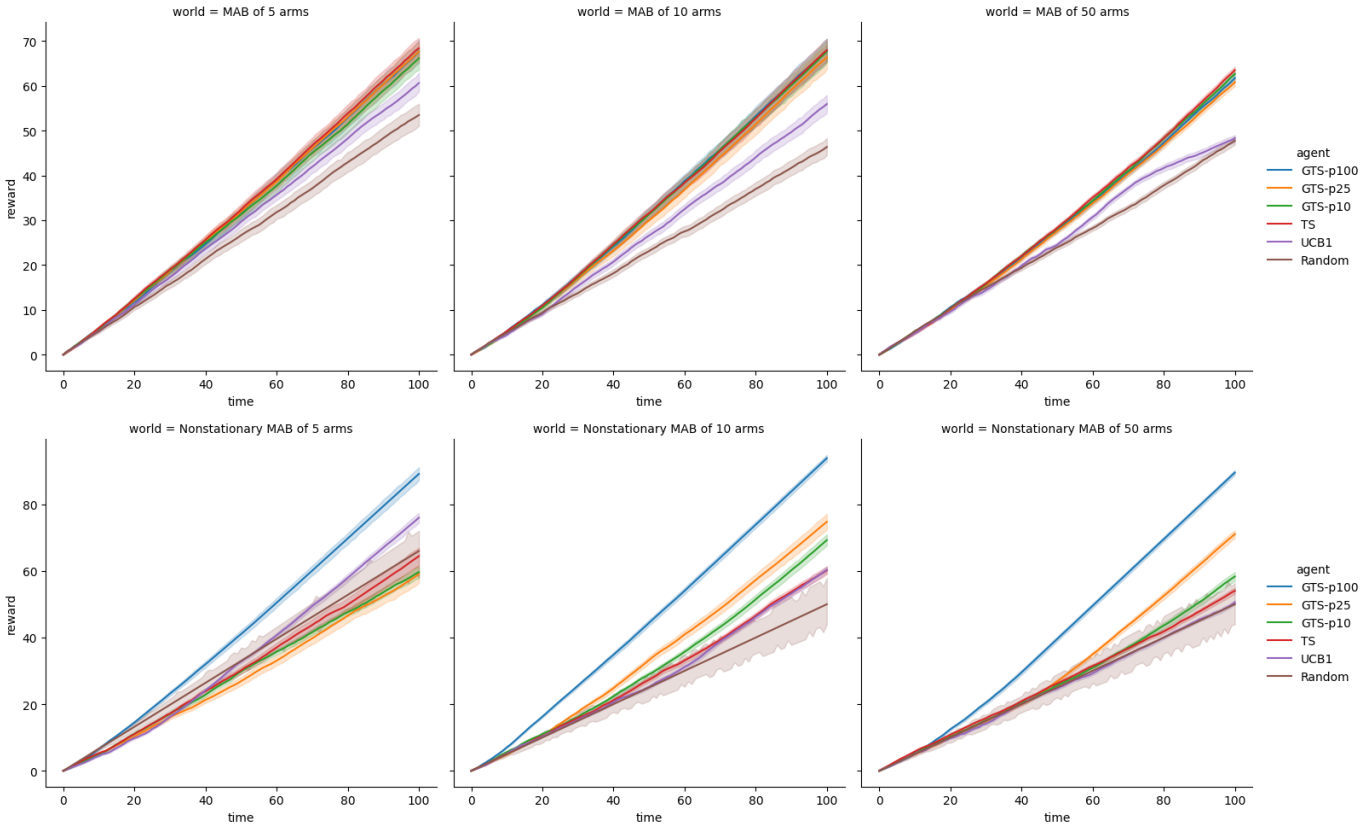

Fig. 1: **Population size.** GTS with bigger population sizes significantly outperforms baselines in nonstationary setting.

the most common one, by choosing the top N agents based on the fitness score. The parameter we use to specify the selection is the selection ratio, and we use 0.5 (i.e. top 50% of agents are kept as elites) throughout the evaluations ahead.

B. The crossover module

The crossover steps are as following. For each freed out spaces for new children, we randomly select two parents from the elite pool. Then, for this new agent, for each bandit arm, we randomly pick one of the two parents, and copy their bandit parameters of that arm as the bandit parameters of that arm for the child. After finishing creating the bandit parameters for this child, we initialize its fitness parameters to $S_f = 1, F_f = 1$, because we have no knowledge of how well this agent will perform just yet. (An alternative would be to use a weighted parameters that is computed as a superposition between the two parents. This would be left for future work to explore.)

C. The mutation module

The mutation models are as following. We first set a mutation rate to indicate how many mutations we want to have in this model. The higher the number, the more mutations the model will introduce. Then, for each mutation times, we randomly pick a agent from the population pool, randomly pick an arm, and then randomly assign a value from -1 to 1 to the bandit parameters of that arm.

VI. Results

Empirically, we evaluate the Genetic Thompson Sampling algorithm in four settings: (1) the bandits with a stationary reward function; (2) the bandits with a nonstationary reward function; (3) an ablation study of different genetic algorithm components in Genetic Thompson Sampling. We report the cumulative rewards of each agent over the learning iterations; and (4) a real-world application of the epidemic control.

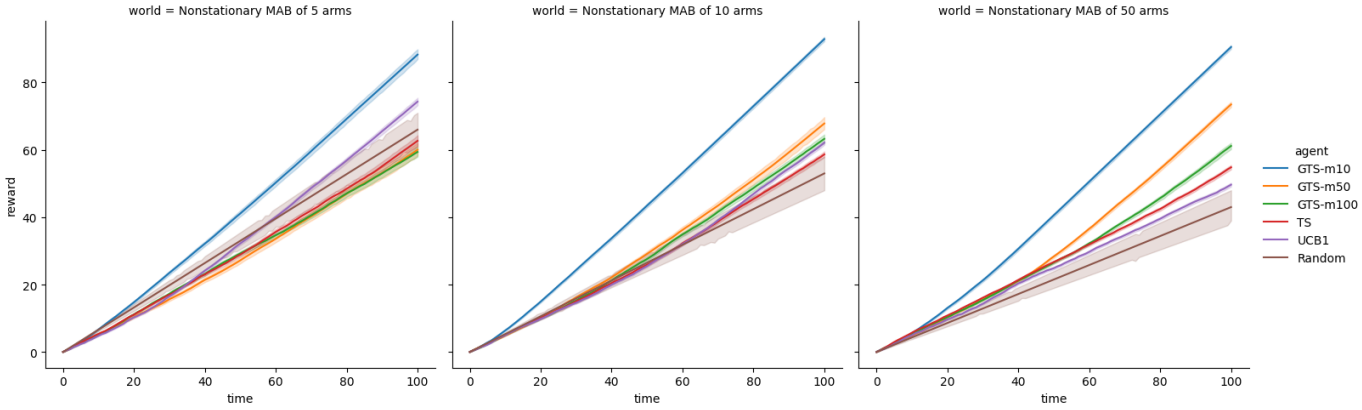

Fig. 2: **Mutation rates.** GTS is sensitive to the levels of mutation, but demonstrates consistent advantages.

A. Multi-armed bandit environment

Simulation environments. We first evaluate the algorithm in a simulated environment of Bernoulli multi-armed bandits. In our simulation, we randomly assign K action arms each with a different probability of giving a reward of 1 or 0. In nonstationary environments, we change the reward distribution every n rounds. We use $n = 10$ throughout the evaluation.

Baselines and variants. We have three baselines. *Random* is a random agent that picks a random action each round. Upper Confidence Bound, or *UCB1*, [42] and the Thompson Sampling, or *TS* [18], are the two theoretically optimal solutions. We have two series of variants of the Genetic Thompson Sampling. (1) In the first evaluation, we test the effect of the population size to the agent performance. For instance, we denote the agent as *GTS-p100* if the population size is 100. In this evaluation, we set the mutation rate to be 10. (2) In the second evaluation, we test the effect of the number of mutations to the agent performance. For instance, we denote the agent as *GTS-m50* if the agent randomly applied 50 mutations to its population. In this evaluation, we set the population size to be 100.

Experimental setting. In our simulation, we randomly generate multiple instances of the environments and randomly initialize multiple instances of the agents. In each world instance, we let the agents make decisions for 100 steps and reveal the reward and cost at each step as their feedbacks. For all the evaluations, there are at least 50 random trials for each agent and we report their mean and standard errors.

Results. Table I and Figure 1 summarize these results. We note that in stationary settings, the Genetic Thompson Sampling is as good as the Thompson Sampling, making the top 2 in all three scenarios. In nonstationary settings, the Genetic Thompson Sampling significantly outperforms the baselines. By varying the population size, we notice that the bigger the population size, the better. The mutation rate is more complicated, we observe that a mutation rate of 10 performs for the population size of 100 (Figure 2.

Ablation study. We perform an ablation study on different components of the genetic updates. We denote having the crossover or not with $C+$ and $C-$, and having the mutation or not with $M+$ and $M-$. As in Table II, both the crossover and mutation contribute to the performance boost, but they don't account for all. Having a majority voting mechanism itself helps the learning of the Genetic Thompson Sampling.

TABLE II: **Ablation study** of the components of genetic algorithm in GTS (presented metric is the cumulative reward).

	NS MAB-5	NS MAB-10	NS MAB-50
TS	57.83 ± 1.30	57.24 ± 1.10	50.79 ± 0.53
GTS (C-, M-)	61.30 ± 1.35	57.72 ± 0.89	54.35 ± 0.57
GTS (C+, M-)	59.39 ± 1.42	63.27 ± 1.28	61.17 ± 0.96
GTS (C-, M+)	60.04 ± 1.93	67.83 ± 1.78	73.49 ± 0.80
GTS (C+, M+)	$\mathbf{88.29 \pm 1.54}$	$\mathbf{92.89 \pm 0.64}$	$\mathbf{90.54 \pm 0.55}$

B. The epidemic control problem

In this evaluation, we consider the practical problem of prescribing intervention plans during a global pandemic. This is an important real-world problem, considering how the COVID-19 has affected the lives of millions of families. The problem is as follows: say, you are a government officer, and you have at hands a series of intervention options. These options are like action dimensions: You can limit the school to two days a week, or you can close the traffic to level 3. For each action dimension, you can have different choices: say, for the traffic control dimension, you can limit it to no closure, level 1 closure (only essential traffic allowed), level 2 closure (only public transport allowed), or level 3 closure (forbid all traffics). However, each choice in each action dimension has a cost. If you close all the traffic, you may stop the spread of the virus, but the economy might crash and people might lose jobs. This is characterized by a stringency value, that evaluates how tight the government resources are. Therefore, the goal is to optimize for two objectives, to minimize the positive cases of the epidemic and to minimize the stringency level of the governmental resources. [8] studied this question first and proposed a bandit solution for it. [8] also introduced a simulation environment that can simulate different epidemic control scenarios and evaluate online learning algorithm.

Simulation environments. As described in [8], in the simulation environment, the user can randomly identify K

action dimension, and randomly identify N different action levels for each action dimension. For instance, the user might design an epidemic control world where there are two action dimensions (i.e. $K = 2$), traffic control and school closure; traffic control can have two levels (degree 1 and degree 2, i.e. $N^{traffic} = 2$) and school closure can have three levels (all schools, all schools except universities, or all primary schools, i.e. $N^{school} = 3$). The user can either consider the budget used by each intervention to be independent (each intervention approach and its value yields a fixed amount of cost regardless of other action dimensions) or combinatorial (the cost of each intervention approach and its value depends on what the action values are in other action dimensions). In real-world, the cost for each intervention approach is usually independent from other intervention dimensions. Thus, we adopt an independent assumption for these cost weights (or stringency weights as in epidemic control terms). As introduced above, the policy makers (i.e. our agents) have access to these stringency weights and thus can use them as contexts in this sequential decision making task. The epidemic control environment here can be nonstationary (which is more realistic), and it is actualized by resetting the weight matrix that maps the each action dimension and choice to the reward distributions and cost distributions. The agent receives a combined feedback of a reward and a cost. To learn from this dual signals, the agents can combine the two feedbacks into one reward given by $r^*(t) = r(t) + \frac{\lambda}{s(t)}$.

Baselines. We evaluate four epidemic control agents. First, we have two random agents. The *Random* agent randomly pick an action value in every action dimension in each decision step. The *RandomFixed* agent randomly picks an action value in each action dimension at the first step, and then stick to this combinatorial intervention plan till the end. Then we have the state-of-the-art agent in this benchmark, the Contextual Combinatorial Thompson Sampling with Budget, or *CCTSB*. For our agent, since our agent is only a multi-armed agent and has no contextual representation installed, we simply use it as a backbone for the combinatorial bandit problem. To be more specific, we can consider each action dimension as its own independent multi-armed bandit (an assumption that is not held in the ground truth). Then we stack these K bandit agents for the K action dimensions together, to form a combinatorial bandit, which we call *IndComb-GTS* where "IndComb" stands for "independent combinatorial" and GTS is its backbone.

Evaluation metrics. To evaluate the problems, we report four metrics. The reward and cost are the ones recorded by the artificial environments. To better match the realistic problem of epidemic control. We post-process these two measures to create two additional metrics corresponding to real-life measures. The "cases" is an estimate of the number of active cases that is infected by the disease, given by an exponential function of the reward: $cases = e^{-reward^*}$, where $reward^*$ is the quantile-binned normalized reward. The "budget" is a quantile-binned metric of the cost. The pareto frontier would be a curve of the number of cases over the used budget.

Results. We evaluate the agents in three scenarios. In the first scenario, we set the λ to be 1, such that the agents are purely driven by the reward. As shown in Figure 3, comparing to the baselines, our agent IndComb-GTS significantly reduces the number of infected cases (and yields the highest rewards), which suggests that it effectively controls the epidemic spread.

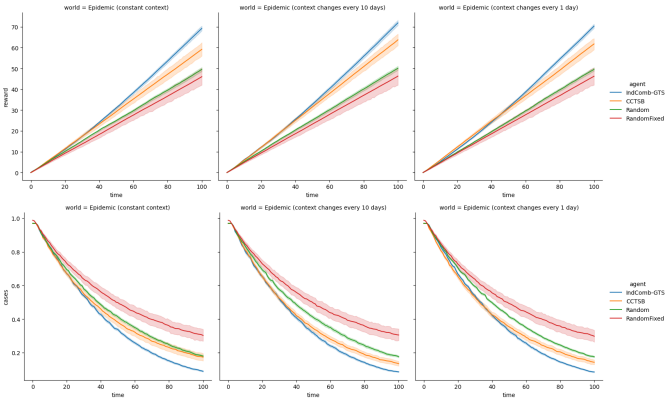

Fig. 3: Reward-driven study: GTS improves case drop.

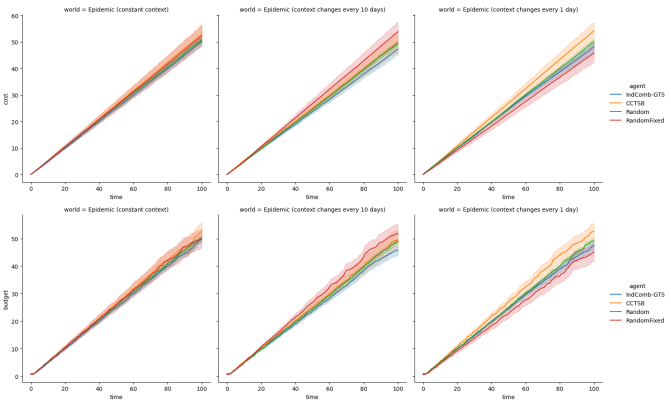

Fig. 4: Cost-driven study: GTS constrains resource stringency.

In the second scenario, we set the λ to be 0, such that the agents are purely driven by the cost. As shown in Figure 4, we see that it yields a similar performance with the state-of-the-art model in reducing the cost and budget. To further evaluate whether GTS can balance the tradeoff between two objectives, we perform a pareto optimal analysis for these agents.

To obtain a pareto frontier for the agents in the epidemic simulation, we run the above evaluations with different values of λ ranging from (0,0.25,0.5,0.75,1). Then we quantile-binned the metrics for each agent and plot out their average and standard errors. As shown in Figure 5, our proposed algorithms yield the pareto optimal frontier, that every intervention plan extracted on its curve will minimize both the number of infected cases in a given day and the resource budget on the government.

VII. The EvoBandit Visualization System

Here we also present EvoBandit, a web-based demonstration system to facilitate the understanding of the Genetic Thompson Sampling. As shown in the screenshots of the system (Figure 6), the user starts off by selecting a proper population size, the number of arms in this multi-armed bandit problem, and

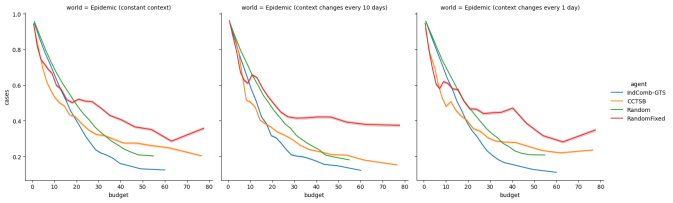

Fig. 5: Pareto frontiers of the cases vs. budget in epidemic.

critical parameters for the genetic algorithm component of the algorithm, such as mutation rates and elite selection ratio. We limit the population size and number of arms only to a limited range, such that the users won't be distracted by too many agents and bandit and lose the main grasp of how the algorithm work. After selecting these system configurations, a grid-like representation of the evolutionary bandit agent is shown with each row corresponding to each arm of the bandts, and each column corresponding to each Thompson Sampling agent in the agent population. Each cell consists of two bars, one for S and one for F (as denoted in Algorithm 4). As in the algorithm, they are each initialized with a random number (same for S and F for each arm within this agent), proportional to the length of the bars on this interface. As these S and F will increase indefinitely during the online learning process, they are rescaled properly to fit the page and only reflect their relative size among each arm and each agent. A fitness is displayed below the grid, corresponding to each agent. On the bottom right corner, there is a printout message board to guide the user along the visualization process. It contains information such as learning step, average reward, and the current stage.

When the user has input their system configurations, they can click "start" to begin the visualization journey. They can also pause and reset the environment any time point in the visualization. There are a few critical states. At each round, we see the agents in the population all make their recommendations, and these recommendations are labeled red. The majority of these recommendations are recorded by marking the bandit arm red. The reward is then revealed, model is then updated and the fitness scores are recomputed. Then in the selection stage, the lowest ranking agents with respect to their fitness score are eliminated, as in their pinked-out stances in the interface. During the crossover, for each new agent to fill the space, two parent agents from the elite pool are randomly selected (marked blue), and their "DNA" (in this case, the parameters for each arms in this agent) are mixed by randomly selecting one parameter set from one parent for each bandit arm (marked blue as well). After all new "children" are introduced into the population, he mutation step (not shown in Figure 6) is simply adding a small number to the parameters of a number of randomly selected agents in the population.

Through this interactive visualization, we believe the users can better understand the learning process of our hybrid model of bandit and genetic algorithm and get interested in pursuing this exciting growing field of research.

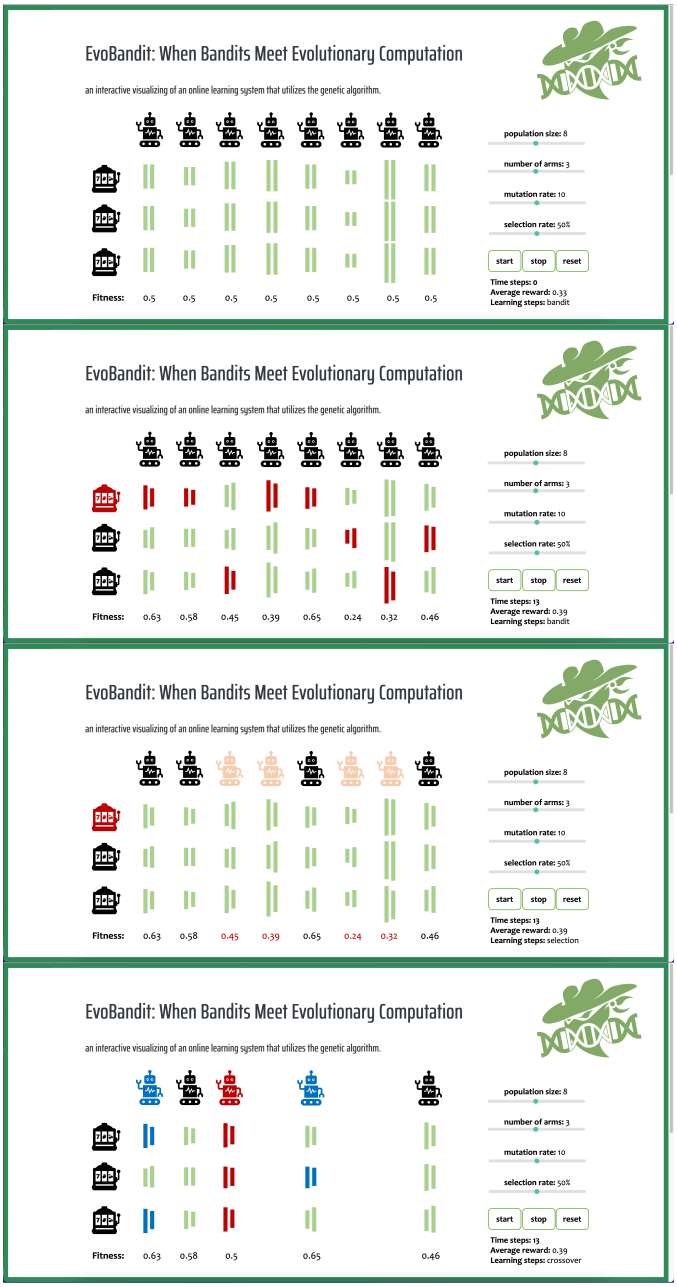

Fig. 6: **EvoBandit Visualization.** Shown here is several screenshots of the web application at different learning states.

VIII. Conclusion

In summary, we propose a hybrid online learning framework that combines the update principles of the genetic algorithm to a bandit algorithm. In the simulation environments of multi-armed bandits and epidemic control, this marriage between evolutionary computation and online learning algorithms appears to be a successful one in nonstationary setting. This study suggest that evoluationary components can be beneficial to the bandit learning problem and worth further investigation. Future work include extending this evolutionary bandit framework to contextual bandits, distributed systems and complex tasks.

REFERENCES

[1] T. L. Lai and Herbert Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[2] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, pp. 235–256, 2002.

[3] Djallel Bouneffouf, Irina Rish, and Charu Aggarwal, "Survey on applications of multi-armed and contextual bandits," in *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2020, pp. 1–8.

[4] Baihan Lin, Guillermo Cecchi, Djallel Bouneffouf, Jenna Reinen, and Irina Rish, "Unified models of human behavioral agents in bandits, contextual bandits and rl," *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) Workshop*, 2020.

[5] Baihan Lin and Xinxin Zhang, "VoiceID on the fly: A speaker recognition system that learns from scratch," in *INTERSPEECH*, 2020.

[6] Baihan Lin and Xinxin Zhang, "Speaker diarization as a fully online learning problem in minivox," *arXiv preprint arXiv:2006.04376*, 2020.

[7] Baihan Lin and Xinxin Zhang, "Speaker diarization as a fully online bandit learning problem in minivox," in *Asian Conference on Machine Learning*. PMLR, 2021, pp. 1660–1674.

[8] Baihan Lin and Djallel Bouneffouf, "Optimal epidemic control as a contextual combinatorial bandit with budget," in *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2022.

[9] Baihan Lin, Guillermo Cecchi, Djallel Bouneffouf, Jenna Reinen, and Irina Rish, "Models of human behavioral agents in bandits, contextual bandits and rl," in *International Workshop on Human Brain and Artificial Intelligence*. Springer, 2021, pp. 14–33.

[10] Kenneth De Jong, "Evolutionary computation: a unified approach," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, 2016, pp. 185–199.

[11] Xin Yao, *Evolutionary computation: Theory and applications*, World scientific, 1999.

[12] Hugh M Cartwright, *Applications of evolutionary computation in chemistry*, vol. 110, Springer Science & Business Media, 2004.

[13] Praveen Ranjan Srivastava and Tai-hoon Kim, "Application of genetic algorithm in software testing," *International Journal of software Engineering and its Applications*, vol. 3, no. 4, pp. 87–96, 2009.

[14] Jany Belluz, Marco Gaudesi, Giovanni Squillero, and Alberto Tonda, "Operator selection using improved dynamic multi-armed bandit," in *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, 2015, pp. 1311–1317.

[15] Jialin Liu, Diego Pérez-Liébana, and Simon M Lucas, "Bandit-based random mutation hill-climbing," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 2145–2151.

[16] Simon M Lucas, Jialin Liu, and Diego Perez-Liebana, "The n-tuple bandit evolutionary algorithm for game agent optimisation," in *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2018, pp. 1–9.

[17] Xin Qiu and Risto Miikkulainen, "Enhancing evolutionary conversion rate optimization via multi-armed bandit algorithms," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 9581–9588.

[18] William R Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[19] Peter Auer and Nicolò Cesa-Bianchi, "On-line learning with malicious noise and the closure algorithm," *Ann. Math. Artif. Intell.*, vol. 23, no. 1-2, pp. 83–99, 1998.

[20] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM J. Comput.*, vol. 32, no. 1, pp. 48–77, 2002.

[21] Djallel Bouneffouf and Raphaël Féraud, "Multi-armed bandit problem with known trend," *Neurocomputing*, vol. 205, pp. 16–21, 2016.

[22] Olivier Chapelle and Lihong Li, "An empirical evaluation of thompson sampling," in *Advances in neural information processing systems*, 2011, pp. 2249–2257.

[23] W.R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples.," *Biometrika*, vol. 25, pp. 285–294, 1933.

[24] Deepayan Chakrabarti, Ravi Kumar, Filip Radlinski, and Eli Upfal, "Mortal multi-armed bandits," in *NIPS*, 2008, pp. 273–280.

[25] Shipra Agrawal and Navin Goyal, "Analysis of thompson sampling for the multi-armed bandit problem," in *COLT 2012 - The 25th Annual Conference on Learning Theory, June 25-27, 2012, Edinburgh, Scotland*, 2012, pp. 39.1–39.26.

[26] Melanie Mitchell, *An introduction to genetic algorithms*, MIT press, 1998.

[27] Kamolwan Kunanusont, Raluca D Gaina, Jialin Liu, Diego Perez-Liebana, and Simon M Lucas, "The n-tuple bandit evolutionary algorithm for automatic game improvement," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 2201–2208.

[28] Kamolwan Kunanusont, Simon Mark Lucas, and Diego Perez-Liebana, "Modeling player experience with the n-tuple bandit evolutionary algorithm," in *Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2018.

[29] Alvaro Fialho, Luis Da Costa, Marc Schoenauer, and Michele Sebag, "Analyzing bandit-based adaptive operator selection mechanisms," *Annals of Mathematics and Artificial Intelligence*, vol. 60, no. 1, pp. 25–64, 2010.

[30] Ke Li, Alvaro Fialho, Sam Kwong, and Qingfu Zhang, "Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 114–130, 2013.

[31] David L St-Pierre and Jialin Liu, "Differential evolution algorithm applied to non-stationary bandit problem," in *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2014, pp. 2397–2403.

[32] Shahin Shahrampour, Alexander Rakhlin, and Ali Jadbabaie, "Multi-armed bandits in multi-agent networks," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 2786–2790.

[33] Timothy Verstraeten, Eugenio Bargiacchi, Pieter JK Libin, Jan Helsen, Diederik M Roijers, and Ann Nowé, "Multi-agent thompson sampling for bandit applications with sparse neighbourhood structures," *Scientific reports*, vol. 10, no. 1, pp. 1–13, 2020.

[34] Abishek Sankararaman, Ayalvadi Ganesh, and Sanjay Shakkottai, "Social learning in multi agent multi armed bandits," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 3, pp. 1–35, 2019.

[35] Baihan Lin, Djallel Bouneffouf, and Guillermo Cecchi, "Online learning in iterated prisoner's dilemma to mimic human behavior," *arXiv preprint arXiv:2006.06580*, 2020.

[36] Abhimanyu Dubey et al., "Cooperative multi-agent bandits with heavy tails," in *International Conference on Machine Learning*. PMLR, 2020, pp. 2730–2739.

[37] Abhimanyu Dubey et al., "Kernel methods for cooperative multi-agent contextual bandits," in *International Conference on Machine Learning*. PMLR, 2020, pp. 2740–2750.

[38] David J Miller and Hasan Uyar, "A mixture of experts classifier with learning based on both labelled and unlabelled data," *Advances in neural information processing systems*, vol. 9, 1996.

[39] Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader, "Twenty years of mixture of experts," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 8, pp. 1177–1193, 2012.

[40] Baihan Lin, Djallel Bouneffouf, Guillermo Cecchi, and Irina Rish, "Contextual bandit with adaptive feature extraction," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2018.

[41] Baihan Lin, "Online semi-supervised learning in contextual bandits with episodic reward," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2020, pp. 407–419.

[42] T. L. Lai and Herbert Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.