




Comparing diverse planning strategies with continuous Monte Carlo Tree Search applied to hybrid Gene Regulatory Networks

1st Romain Michelucci 
I3S, CNRS
Université Côte d’Azur
Sophia Antipolis, France

2nd Jean-Paul Comet 
I3S, CNRS
Université Côte d’Azur
Sophia Antipolis, France

3rd Denis Pallez 
I3S, CNRS
Université Côte d’Azur
Sophia Antipolis, France

Abstract—Real-world applications of artificial intelligence often require the decision-maker to choose between multiple optimal solutions at hand before making a final decision. Since there is no guarantee that an increase in the budget or several independent executions will yield different solutions, classic mechanisms are not suitable for identifying multiple solutions. In the context of sequential decision-making problems, Monte Carlo Tree Search (MCTS) is a state-of-the-art online planning algorithm. It is responsible for the improvement of many computer games but also for real-world problems involving continuous action spaces. MCTS has recently been successfully applied to the diverse planning problem in the discrete setting. In this work, we propose different diverse planners based on MCTS (DP-MCTS) to be relevant in the continuous setting. The solution to a diverse planning problem is a Pareto set between diversity and quality of plans. Therefore, we suggest considering a multi-objective setting in which the vectorial reward integrates the diversity measure as an additional objective. In addition, we propose two types of inhibition strategies disregarding the optimal plans to enforce the exploration of the search space during the tree construction. The three different contributions are assessed independently against a diverse multi-armed bandit policy, and the comparison is held on a real-world biological problem involving continuous action and state spaces.

Index Terms—continuous MCTS, diverse planning, inhibition strategies, diverse multi-objective, chronotherapy, hybrid GRN

I. INTRODUCTION

Reinforcement learning [1] addresses sequential decision problems in the Markov Decision Process (MDP) framework. In this context, MCTS is a search method for finding optimal decisions by taking random samples in the decision space and building a search tree according to the statistics obtained. In such trees, nodes denote states, whereas edges represent actions leading from one state to another. MCTS has not only been designed for making computer players in Go [2], it also proved its effectiveness in a wide variety of settings, including General Game Playing (GGP), and is not limited to games [3], [4]. The most popular algorithm is Upper Confidence bounds applied to Trees (UCT) [5] using the UCB formula [6] as a selection function that aims at maintaining a proper balance between the *exploration* of not well-tested actions and *exploitation* of the best actions identified so far. The *all-moves-as-first* (AMAF) [7], [8] enhancement heuristic

updates statistics for all actions selected during a simulation as if they were the first action applied. Rapid Action Value Estimation (RAVE) [9], [10] is a generalized idea of AMAF to search trees where action values are shared among subtrees. GRAVE [11] generalizes the RAVE algorithm to have more accurate estimates near the leaves. The resulting algorithm outperformed RAVE on multiple games, such as Go, Knight-through, and Domineering, without any specific knowledge.

All these successful variants are designed for finite and discrete action spaces. They require trying every action once, which is impossible in continuous spaces. Recent advances address this limitation: progressive widening [12], [13] (PW), also known as progressive unpruning [14] can handle continuous action spaces by considering a growing set of sampled actions. cRAVE [15] combines the PW strategy with a modification of the RAVE heuristic to do information sharing by generalizing from similar actions and states. Recently, cGRAVE [16] has been proposed to expand the GRAVE heuristic in the continuous action and state spaces. Many real-world problems involve selecting a sequence of actions, i.e., a plan, from a continuous space of actions. Examples of continuous applications are control tasks in OpenAI Gym environment [17], robotic planning [18], action selection in an Olympic Curling simulator [19], and identifying parameters of hybrid gene regulatory networks [16].

On top of that, in some real-world applications, a decision-maker prefers to seek multiple solutions at hand before making a final decision. If one solution is not suitable, an alternative can be proposed immediately rather than being pursued iteratively. The *Second Toyota Paradox* [20] is a well-known example illustrating the interest of such a strategy. The goal of searching for multiple solutions in a single run differs from the traditional single-solution-seeking mechanism. Indeed, it is by no means guaranteed to produce different solutions across multiple runs or by simply increasing the budget.

In the context of classical planning problems, the generation of sets of plans has been extensively studied in previous research such as *top-k* [21], *top-quality* [22] and *diverse planning* [23]–[25]. In the context of offline and single-player sequential decision problems, in which no symbolic model of

the problem instance is required, this area has been recently covered in [26] for discrete state spaces.

In this paper, we present three different contributions to the one presented in [26], where we tackle the problem of diverse planning in a continuous setting. We exhibit diverse and top-quality continuous plans on the problem of identifying dynamic parameters of gene regulatory networks (GRNs). The contributions of this paper are (i) the formulation of diverse planning with MCTS as a multi-objective problem and (ii) two different inhibition strategies enforcing diversity during the construction of the search tree. Each of the three resulting diverse planners is assessed independently against a *diverse multi-armed bandit policy* introduced in [26].

The paper is organized into three remaining sections: Section II introduces notations and related background. Section III describes the proposed DP-MCTS variants. Section IV presents the experimental validation of the different contributions to a real-world biological problem.

II. BACKGROUND

A. Deterministic Markov Decision Processes

In the context of deterministic single-agent planning, problems are usually formulated as a discrete-time and finite MDP. In this setting, the agent and environment interact at each of a sequence of discrete timesteps $t \in \llbracket 1, n \rrbracket$. At each timestep, the agent receives some representation of the environment's state s_t and, in accordance with this information, selects an action a_t . Consequently, in the next timestep, the agent receives a reward r_{t+1} and finds itself in a new environment state s_{t+1} . A *trajectory* is the sequence of state, action, and reward chosen by the agent: $s_0, a_0, r_1, s_1, a_1, r_2, s_2, \dots, s_l$ with s_l being a terminal state. An MDP is characterized by a tuple (S, A, T, R) where S is the state space and A is the action space, $A(s)$ is the action space given a state s . Assuming a deterministic transition function, $T : S \times A \rightarrow S$ gives the next state s_{t+1} , knowing the action a_t and the state s_t . The reward function $R : S \times A \rightarrow \mathbb{R}$ gives the immediate reward after transitioning from state s_t to s_{t+1} . In a deterministic MDP, a *plan*, denoted π , is a sequence of actions leading to a terminal node $\pi = (a_0, \dots, a_l)$. An *optimal plan* is a sequence of actions that maximizes the cumulative reward.

B. Monte-Carlo Tree Search

a) *Discrete MCTS*: The basic algorithm involves iteratively building a search tree until some predefined computational budget is reached (usually time, memory, or iteration). An action a_t is represented as an edge starting from s_t to s_{t+1} , both depicted as nodes. Four steps are applied for each iteration search:

- Selection: starting at the root node s_0 , a node *selection function* is recursively applied to descend through the tree until a non-terminal leaf node is reached,
- Expansion: one or more child nodes are added to the built tree according to the set of legal and available actions. Along with the selection step, they form the *tree policy*,

- Rollout: a simulation is played from the expanded node according to the *default policy* (usually, random actions) until a terminal state s_l . This step results in a cumulative reward,
- Backpropagation: the cumulative reward is backed up through the nodes selected in the first step, and the statistics are updated. These statistics are used in the selection function for future decisions.

In the baseline MCTS, an action is uniformly sampled from the action space. When dealing with a large action space, this raises the problem of exploration *versus* exploitation trade-off. The Upper Confidence bound for Trees (UCT) treats this issue as a multi-armed bandit problem with the Upper Confidence Bound (UCB): the value of each action is their expected reward calculated by the Monte Carlo simulations, and the rewards are the random variables with unknown distributions. UCT chooses the action $a_t \in A(s)$ from s_t with the UCB formula:

$$\operatorname{argmax}_{a_t} (\operatorname{mean}_{a_t} + c \times \sqrt{\frac{\log(N(s_t))}{N(s_t, a_t)}}) \quad (1)$$

where $\operatorname{mean}_{a_t}$ is the empirical mean of rewards obtained when a_t was chosen during the selection step, $N(s_t)$ the number of times the state s_t has been selected, $N(s_t, a_t)$ the number of times the action a_t has been chosen after s_t was selected, and $c \in \mathbb{R}^+$ is the exploration parameter that balances exploration and exploitation (tuned for each problem).

AMAF is an enhancement of UCT. The strategy takes place during the backpropagation step. The statistics of the actions chosen during the selection step are updated, as well as the ones involved in the default policy. The name *all-moves-at-first* comes from the fact that the actions chosen during a simulation are treated as if they were the first action applied.

RAVE is a popular AMAF enhancement that blends the UCT score with the AMAF score. The idea is to minimize the problem of early estimations: when there are not many samples, RAVE aims at a more robust assessment of actions by sharing the rewards gathered along different subtrees of the search tree. To select an action a_t , the RAVE formula involves the AMAF value of the action a_t ($AMAF_{a_t}$):

$$\operatorname{argmax}_{a_t} ((1.0 - \beta_{a_t}) \times \operatorname{mean}_{a_t} + \beta_{a_t} \times AMAF_{a_t})$$

and a weight β_{a_t} :

$$\beta_{a_t} = \frac{pAMAF_{a_t}}{pAMAF_{a_t} + p_{a_t} + \operatorname{bias} \times pAMAF_{a_t} \times p_{a_t}}$$

$pAMAF_{a_t}$ is the number of rollouts containing action a_t , p_{a_t} is the number of rollouts starting with action a_t , and *bias* is a problem-dependent parameter controlling the exploration bias.

The principle of GRAVE is to use AMAF values of a state higher up in the search tree than the current state (commonly named *tree*). Even if the statistics of an ancestor state refer to some actions made earlier, a state upper in the tree has better accuracy because it has more associated rollouts. GRAVE shows how using ancestor statistics when the number of

simulations is too low is more accurate than using statistics of the actual node.

b) *Continuous MCTS*: A fundamental assumption for applying these selection functions is that they require each action to be tried at least once. They are not applicable in cases where the cardinality of the action space is very large with respect to the number of iterations ($|A| \gg n$). Progressive widening (PW) is a widely employed solution limiting the number of actions of a state s_t based on its number of visits (restricted by the power of pw). A new action a_t is sampled from s_t each time the visitation counter of s_t ($N(s_t)$) is greater than or equal to its number of actions :

$$N(s_t)^{pw} \geq |C(s_t)| \quad (2)$$

where pw is a problem-dependent parameter that restrains the number of actions allowed in s_t , and $C(s_t)$ is the set of actions from s_t ($C(s_t) \subset A(s_t)$).

While the selection function ensures that the tree grows deeper in the promising regions of the search space by balancing exploration and exploitation, the PW strategy guarantees that it grows wider in those regions.

cRAVE extends RAVE to the continuous action and state spaces. As stated in [15], the state-action values are smoothly estimated thanks to Gaussian convolutions: every tree-walk in the subtree of s comprising state-action pair (s_i, a_i) is considered with a weight decreasing exponentially depending on (i) the distance between the executed action a_t and the considered action a_i , and (ii) the distance between s_i and s_t .

In the same idea, cGRAVE has been recently proposed to build upon GRAVE by using the smoothed state and action values estimations of ancestors to increase the accuracy of the estimates in leaf nodes.

C. Diverse planning with MCTS

Generating multiple solutions, instead of a single one, in a planning problem is called *top-k*, *top-quality* or *diverse* planning. The choice of the planner depends on the characteristics of the solutions set the user is looking for. Top-k planning addresses the problem of finding a finite set of the k-best plans according to some quality measure. Top-quality imposes that the non-predetermined set of plans has a minimum quality value. Diverse planning adds to the top-quality planner a constraint on the plan similarity: it enforces obtaining both the maximum quality and the most diversity between the plans. As we are interested in diverse planning, only this type of planner will be presented in detail below.

Diverse planning addresses the problem of generating sets of plans that are significantly different according to some distance measure. Significantly means that the set of plans must be distant from one another with respect to a distance threshold. Diversity is commonly defined as the average or minimum pairwise distance within the set of plans. But it can also be based on qualitative (specific to the domain) [27], quantitative [25], [28] (domain-independent) measures, or both [29].

Following the definition of diverse planning in [26], we consider the problem of maximising the quality of plans given a bound on diversity :

Definition 1 (Diverse top-k-quality planning problem). *Given a planning problem (S, A, T, R) , natural number k , measure of plan quality $Q_{plan}(\pi)$, quality constraint q , measure of diversity $D(\pi, P)$ between a plan π and a set of plans P , and a distance threshold d , find a subset P in the set of all plans P_π such that:*

- *there exists no plan $\psi \in P_\pi$ s.t. $\psi \notin P$ having diversity $D(\psi, P) \geq d$ and quality $Q_{plan}(\psi) \geq Q_{plan}(\pi)$,*
- *$\forall \pi \in P, D(\pi, P \setminus \{\pi\}) \geq d$*
- *$\forall \pi \in P, Q_{plan}(\pi) \geq q$,*
- *$|P| \leq k$*

Thus, a solution to a diverse planning problem is a set of plans P in which every plan is quality-optimal and P is diverse. In other words, it is a Pareto set balancing diversity and quality.

In the context of planning, MCTS is used to generate plans by extracting action sequences from the search tree (a plan π corresponds to any action path from a_0 to an action leading to a terminal node a_l). In the simplest case, generating an optimal plan consists of recursively selecting the child node with the maximum value until a terminal node is reached. However, considering a generation of a set of plans requires comparing the alternative plans of a search tree that, for instance, might be of different lengths. And, unlike classical planning, in standard MCTS, rewards are assumed to be received only in terminal states. It is the main reason why a *plan quality metric* (Equation (3)) and a *diverse plan extraction algorithm* have been introduced in [26]. The plan quality metric is defined as the product of the regret of suboptimal action choices. It helps to evaluate a plan relative to the optimal one and allows one to compare plans with different lengths:

$$Q_{plan}(\pi) = \prod_{i=0}^{n-1} \frac{mean_{a_{i+1}}}{\max_{a' \in C(a_i)} mean_{a'}} \quad (3)$$

The plan extraction algorithm takes as input a search tree and outputs a set of optimal and diverse plans. It iteratively collects and extends *plan stems* until a plan has reached a terminal node. A plan stem π_{a_t} is a sequence of actions from a_0 to the current action a_t (which does not lead to a terminal node). A plan is added to the set of top plans if both quality and diversity thresholds are satisfied. The algorithm stops when the desired number of plans k has been generated. Therefore, the algorithm guarantees generating a set of plans such that no plan of greater quality exists outside of the returned set.

On top of that, *diverse multi-armed bandit policies* (DMAB) [26] in MCTS have been introduced to search online a diverse set of plans. The tree policy favours the actions with the greatest potential value. Usually, this value is given by the heuristic chosen, such as UCT, RAVE, or GRAVE. In DMAB, the authors suggest biasing the selection function to promote

potentially good actions that are distant from the set of actions saved in P :

$$\operatorname{argmax}_{a_t \in A(s_t)} (UCB(a_t) + D(\pi_{a_t}, P)) \quad (4)$$

where $UCB(a_t)$ relates to Equation (1) and $D(\pi_{a_t}, P)$ is the diversity of the plan stem π_{a_t} to the saved set of plans P .

III. CONTRIBUTIONS

This section presents three new DP-MCTS variants dealing with the problem of collecting diverse plans during a single execution in a continuous or large-scale domain.

A. Inhibition strategies

First, we propose two strategies (Lock and Diverse PW) enforcing the diversity of the final set of top-quality and diverse plans P during the construction of the search tree. The underlying idea of such strategies is to inhibit areas of the search space that already lead to optimal plans, to reinforce the exploration of promising areas. We call these strategies inhibition strategies.

a) Lock: The first approach is simply to lock tree branches in which a complete plan, a solution to the problem, is stored in P . When a complete plan π is stored in P , its selected actions (a_0, \dots, a_T) are locked. This means that the tree branch (not just the plan) from the search tree is inhibited.

b) Diverse Progressive Widening (DivPW): Our second approach suggests modifying the PW formula (Equation (2)) to promote additional action sampling. We add to PW the constraint that the actions starting from s_t must not belong to the set of actions saved in the complete plans of P at the same depth ($P.actions(t)$). Instead of considering the number of actions ($|C(s_t)|$), we only consider the subset of actions that are not present in any of the complete plans: $C_{DivPW}(s_t) = C(s_t) \setminus \{a_t \mid a_t \in C(s_t) \cap P.actions(t)\}$:

$$N(s_t)^{pw} \geq |C_{DivPW}(s_t)| \quad (5)$$

During the construction of the search tree, this encourages exploration at each depth, especially when a high number of actions is already present. In addition, if the DivPW condition (Equation (5)) is met, the selection function is only applied to the subset of actions that do not belong to any saved plan in P (inhibiting them). When the DivPW condition is not met, we avoid resampling an existing action.

By inhibiting plans or actions, these heuristics encourage exploration of unvisited or less promising areas of the search space.

B. Diversity as an objective

We suggest formulating the problem of diverse planning as a multi-objective reinforcement learning problem. In the DMAB, diversity of solutions primarily serves as a necessary property of online discovery processes and not as an objective in its own right. In fact, with DMAB, diverse planning is mostly regarded as an intermediate problem, being a linear combination between the score and the diversity. Since the

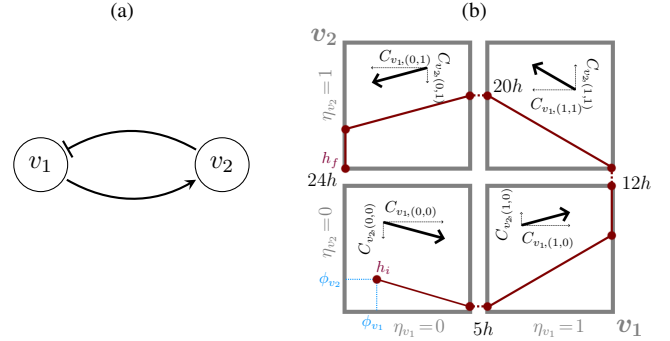


Fig. 1: Example of a hGRN model depicted as a directed graph (a), and a possible hybrid state graph (b). The hGRN continuous parameters are depicted as black arrows.

solution to a diverse planning problem is a Pareto set balancing diversity and quality, we consider using Multi-Objective MCTS [30] (MO-MCTS).

MO-MCTS is the multi-objective extension of MCTS and has been shown to outperform linear-scalarization approaches. Instead of aggregating the objectives into one, MO-MCTS considers a reward vector $r \in \mathbb{R}^d$ where d is the number of objectives. During the search, an archive X maintains all non-dominated vectorial rewards evaluated in previous tree walks. Given X and the newly vectorial reward obtained at the simulation step, the *hyper-volume* indicator [31] (HV) is used to calculate the modified selection function:

$$MO_a = \begin{cases} HV(X \cup \{\bar{r}_{a_t}\}; z) & \text{if } \bar{r}_{a_t} \text{ is non-dominated in } X \\ HV(X \cup \{\bar{r}_{a_t}\}; z) - \|\bar{r}_{a_t}^p - \bar{r}_{a_t}\|_2 & \text{otherwise.} \end{cases}$$

where $z \in \mathbb{R}^d$ is the reference point of HV , \bar{r}_{a_t} is a d -dimensional vector in which the selection function is applied to a_t for each objective, and $\bar{r}_{a_t}^p$ is the perspective projection of \bar{r}_{a_t} onto X .

In this context, we introduce the diversity score as an additional objective: $score = (Q_{plan}(\pi), D(\pi, P))$. The diversity score is calculated using the plan dissimilarity measure provided by the user, such as the minimum distance from any plan in the existing set P . It can also be based on qualitative (domain-specific) or quantitative (domain-independent) measures. In the following, this variant of DP-MCTS will be referred to as DPMO.

IV. APPLICATION

To evaluate our proposed MCTS diverse planners, we consider the problem of identifying dynamic parameters in hybrid gene regulatory networks (hGRNs). Our goal is to generate a bounded set of top-quality plans that maximises the diversity of continuous parameters found, meaning that they are compliant with the biological knowledge (top-quality criterion). This assessment is done on two problem instances: the circadian clock (3G) and the cell cycle (5G).

A. hGRNs parameters identification

Hybrid modelling of GRNs [32] aims to describe the effect of regulations between genes in a biological system taking into account the continuous time component. Usually, a GRN is represented as a directed graph in which vertices express abstractions of one or multiple biological genes having the same effect, and edges act as regulations. Regulations can either be an activation ($\xrightarrow{+w}$) or an inhibition ($\xrightarrow{-}$) of a target vertex, only if the discrete concentration of the source vertex is above its w^{th} threshold (an unlabelled edge means $w = 1$). In the context of Figure 1a, the maximum discrete level of both genes is 1, leading to discrete levels of genes (η_{v_1} and η_{v_2}) in $\{0, 1\}$. This static representation is of limited interest since it does not help the modeller to predict the temporal dynamics of the system. Although a discrete dynamical framework has been developed, we consider here a hybrid modelling framework that adds chronometric aspects to the discrete framework, because it is fundamental to observe and reason not only about the discrete dynamics of a complex system but also about its chronometric evolution. It is particularly important in *chronotherapy* to optimise medical treatments by taking into account biological rhythms.

The hGRN dynamics of Figure 1b can be built in two steps: (i) first, each grey box representing a discrete state defines the discrete concentration level of each gene, then (ii) the hGRN dynamics are defined as piecewise linear continuous trajectories (red lines). The trajectory starts from an initial hybrid state h_i , represented by both a discrete state η and a precise position ϕ inside the discrete state. The initial state is defined by $h_i = ((\eta_{v_1}, \eta_{v_2})^t, (\phi_{v_1}, \phi_{v_2})^t) = ((0, 0)^t, (0.25, 0.25)^t)$. Then, the temporal evolution inside each discrete state is given by a so-called *celerity vector* (black arrows), which defines the direction and celerity of each gene, e.g., the celerity of v_1 in $\eta = (0, 0)$ is denoted $C_{v_1, (0,0)}$. More generally, the celerity of v in η is denoted $C_{v, \eta}$. Identifying a valid set of celerity vectors (dynamic parameters) could help biologists make new interpretations of the possible system dynamics.

We are interested in valid hGRN models of the biological system under study, that is, into hGRN models consistent with knowledge and observations. Our approach takes into consideration already-formalized information analysed by biologists derived from biological data and expertise instead of raw data, which are known to be subject to noisiness and scarcity. The approach abstracts the knowledge extracted from biological experiments under the form of constraints on the global trajectory: it must (i) start from an initial hybrid state $h_i = (\eta_i, \phi_i)$, (ii) verify in each successive discrete state a triplet of properties $(\Delta t, b, e)$ where Δt expresses the time spent; b delineates the observed continuous behaviour inside the discrete state (\top means the absence of observed behaviours); e specifies the next discrete state transition, and (iii) reach a final hybrid state $h_f = (\eta_f, \phi_f)$. For the interaction graph of Figure 1a, biological expertise can be summarized as follows: there exists a behaviour starting from specific coordinates (h_i) going through four discrete states

and finishing at other specific coordinates (h_f) after 24 hours. More precisely, the time spent in each of the four discrete states is approximately 5 hours in $(0, 0)$, 7 in $(1, 0)$, and so on. See the first properties of each event in the following description of the biological knowledge (BK):

$$\{h_i\} \left(\begin{array}{c} 5.0 \\ noslide(v_2) \\ v_{1+} \end{array} \right); \left(\begin{array}{c} 7.0 \\ slide^+(v_1) \\ v_{2+} \end{array} \right); \left(\begin{array}{c} 8.0 \\ noslide(v_2) \\ v_{1-} \end{array} \right); \left(\begin{array}{c} 4.0 \\ slide^-(v_1) \\ v_{2-} \end{array} \right) \{h_f\}$$

For the first triple, v_{1+} constrains the trajectory to reach the next discrete state by increasing the concentration level of v_1 . The second property $noslide(v_2)$ in $(0, 0)$ expresses that the trajectory has to reach the right border of the discrete state without touching the upper or lower borders (expressing the saturation). The continuous trajectory of Figure 1b satisfies all properties of Section IV-A.

Any valuation of dynamic parameters leading to a trajectory satisfying BK is considered a solution to the hGRN identification problem.

B. Experimental validation

a) *Problem specifications and instances*: We consider the problem of identifying continuous parameters in hGRNs as an RL problem, more specifically, as an online decision-making problem modelled as a deterministic MDP. We do not consider classical planning algorithms since (i) no symbolic model of the problem is available (a simulator is used), and (ii) the quality of a plan is not defined by the sum of action costs but is only rewarded in terminal states.

In this MDP, a state s_t is defined as an incoming hybrid state (the first point in each discrete state following the piecewise linear trajectory evolution) and its corresponding time, and an action a_t is a celerity vector. In each timestep, an action is composed of d continuous variables, where d is the number of genes in the system. Since an optimal plan must satisfy every triplet of BK, its length must equal the number of triplets. Thus, a reward for a simulation equals the length of the simulated trajectory (the number of discrete states it goes through) before ending in a terminal state. A terminal state is either a final state (maximum length) or an intermediate state which does not respect the BK constraints (at $\epsilon = 1e-2$). For instance, in the hGRN presented in Figure 1, a reward of 4 means that the plan is optimal because the trajectory satisfies each 4 triplets of BK (described in Section IV-A) and is valid. While a reward below 4 corresponds to a trajectory being stuck (not valid). Details are provided in [16].

The experimental validation is held on two real-world hybrid models of GRN. They are depicted in Figure 2: the circadian clock (with 3 genes, in short 3G) and the cell cycle (with 5 genes, in short 5G). Each hGRN parameters are described in terms of (i) the number of genes d (the cardinality of an action), (ii) the number of actions to find, i.e., the maximum length of a plan, and (iii) constraints from BK utilised for evaluating a simulation.

b) *Experimental setting*: In the following experiments, we test four MCTS diverse planners: DMAB (the baseline), Lock, DivPW and DPMO. They generate a set of plans for

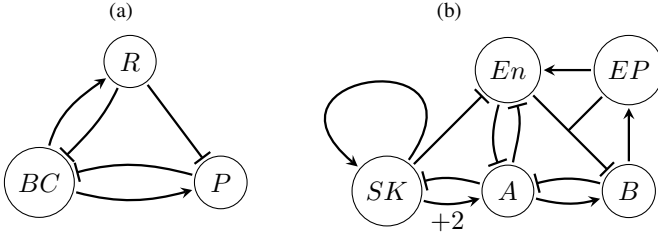


Fig. 2: Interaction graphs of the 3G (a), and 5G (b) hGRN.

name	Nb. genes	Nb. continuous params	BK
circadian cycle (3G)	3	8	[32]
cell cycle (5G)	5	48	[33]

TABLE I: Description of hGRN models.

up to five solutions ($k = |P| \leq 5$), enforce a minimum 100% of the quality of the optimal plan ($q = 1.0$) and a minimum plan distance of $d = 1e - 1$. The distance measure chosen is the Euclidean action distance. This choice has been made since a diverse set of actions always leads to a diverse set of biological traces. A dissimilarity measure based on state distance is not relevant since the measure could assign a similar value for a newly discovered plan even if its sequence of actions would have been different. Thus, the diversity metric when considering a new plan π is the minimum distance from any plan in P :

$$D(\pi, P) = \inf\{\|\pi, \pi'\|_2 \mid \pi' \in P\} \quad (6)$$

We use the cGRAVE algorithm, where GRAVE is the selection function in Equation (4), since previous research [16] showed that it was the most relevant (among cUCT and cRAVE).

The ad-hoc and manual parameter tuning of the algorithms is known to have multiple disadvantages, such as being biased by experience, time-intensive and limited by the number of problem instances. We use an iterated racing procedure for automatic algorithm configuration: the `irace` package [34]. The best elite configuration obtained after 1000 iterations has been kept to determine the values of the problem-dependent parameters. For the cGRAVE heuristics, the parameter space is $bias \in \{1e - 15, 1e - 14, \dots, 1e - 2, 0.1\}$, $\alpha_{state} \in \llbracket 1, 100 \rrbracket$, $\alpha_{action} \in \llbracket 1, 100 \rrbracket$, and $pw \in \{0.3, \dots, 0.89, 0.7\}$. The resulted tuned values are $bias = 0.1$, $\alpha_{state} = 47$, $\alpha_{action} = 81$, and $pw = 0.61$. ref is found among the values $\llbracket 1, 100 \rrbracket$ and equals 15. The Euclidean distance is chosen for action and state spaces. Each experiment is run 30 times to obtain statistically significant results. The diverse planner performances are compared for the same computational budget, i.e., 50.000 tree-walks in 3G and 200.000 in 5G.

c) Results analysis: Firstly, we focus on analysing and thus comparing the diversity of the set of solutions P generated by each DP-MCTS variant. Such performance is assessed thanks to the sum of distances (SD) measure. It finds the

Alg	3G			5G		
	mean \pm std	max	min	mean \pm std	max	min
DMAB	29.33 \pm 3.41	34.9	24.19	93.29 \pm 57.35	225.87	18.87
DPMO	23.12 \pm 3.31	28.36	16.26	97.91 \pm 16.85	115.14	77.02
Lock	33.95 \pm 2.96	39.84	26.77	50.03 \pm 48.68	226.61	15.91
DivPW	34.48 \pm 2.42	39.35	29.59	126.4 \pm 24.92	182.53	87.6

TABLE II: Diversity statistics (Equation (7)) of the set of plans generated by the different planners on both problem instances: 3G and 5G. Bold values denote the best results column by column.

dissimilarity between plans by calculating the square root of the sum of their distances from one another:

$$SD(P) = \sqrt{\sum_{\pi_i \in P} \sum_{\substack{\pi_j \in P \\ j \neq i}} \|\pi_i - \pi_j\|_2} \quad (7)$$

where π_i and π_j are two distinct plans in P .

Table II represents the mean, the standard deviation and the extrema of the SD measurements obtained by each diverse planner on the two problem instances. The best results are bolded. From this table, we can deduce that, both in 3G and 5G, DivPW maintains a more diverse set of solutions, and Lock finds the best one (given the 30 executions). Nevertheless, the naïve strategy employed by Lock is opportunistic: average performances are lower than other planners for 5G. DPMO and DMAB obtain similar statistics in both instances. DMAB shows higher performance variability (visible thanks to the standard deviation).

In the second stage, we focus on a different aspect: what if the set of plans P was not bounded ($k = \infty$ in Definition 1)? In fact, from a modeller's perspective, having the widest choice possible can be instructive and a valuable feature. In addition, the number of solutions is not necessarily known in advance. For instance, in our context, exhibiting the largest solutions sample is interesting for discussing with biologists to identify the most relevant interpretations among the solutions extracted: the larger the number, the better. It is the reason why, during our experiments, a second set of plans P' has been introduced to collect every plan that is both top-quality and different ($D(\pi, P') \neq 0$), but this set was unbounded: it comes down to a top-quality problem where the plans set is only bounded on quality, and we assess the diversity performances of each planner. In this context, we aim at obtaining the largest and most diverse solutions set. We employed the following performance measure, inspired from [35]:

$$sc(P') = |clust_\sigma(P')| \quad (8)$$

The score measurement uses density-based clustering with parameter σ to remove redundancy between plans closely clustered and returns the number of obtained clusters. Each representative plan of the clusters found (with the higher quality value) is kept as an optimal solution, while the others are removed. In this study, OPTICS [36] is parametrised with $\sigma = d = 10^{-1}$.

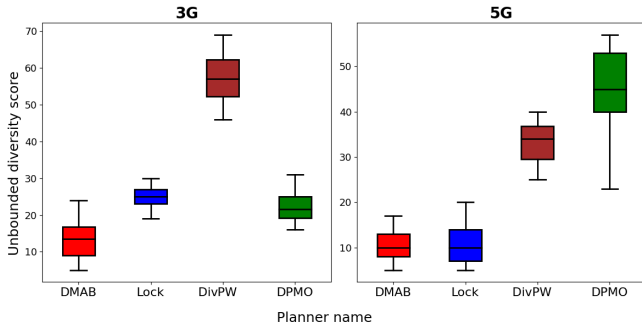


Fig. 3: Boxplots of average diversity score (Equation (8)) generated by each planner. The higher, the better.

From Figure 3, we can state that, in 3G, DivPW largely outperforms other tested planners. DPMO and Lock obtain better diversity performances than DMAB, and their performances are not statistically different. In 5G, the DMAB baseline also lags behind the other proposed strategies. However, the DivPW strategy is surpassed by DPMO. Overall, Figure 3 demonstrates the interest of the newly proposed DP-MCTS in both problem instances when searching for an unbounded diverse set of plans.

d) Discussion: According to our experiments, the two best planners standing out are DivPW and DPMO. As with DMAB, one drawback of such heuristics-based planners is that there is no evidence that the exploration biases the search toward high-quality diverse plans. However, on the tested problem, they have shown to exhibit a set of plans with higher diversity than our baseline DMAB.

Following Figure 3, the results obtained by planners in 5G are lower than the ones displayed in 3G, but they are not comparable since the dimensionality of the action space is different (see Table I). In addition, the problem instance complexity increases as the number of parameters to identify increases exponentially with respect to (i) the number of genes d , and (ii) the concentration levels of genes. Similarly, from Table II, the SD scores naturally increase between the two instances since there are a higher number of actions to identify leading to a larger action space. This also explains the results variability of the different heuristics-based planners.

It must be noted that MO-MCTS is known to suffer from a higher computational cost due to the hypervolume computation (which is then the case with DPMO): the user could prefer obtaining a lower diversity set of plans for the benefit of faster calculation by using DivPW.

C. Visualisation

Figure 4 and Figure 5 show the diversity of the set of solutions found in the set of plans P of each planner tested on the two problem instances. It exemplifies the solution set generated by each planner given a single execution. Please note that two different graph types are modelled to emphasize the same phenomenon: the evolution of gene product concentration. The 3G discrete states (Figure 4) can be represented as

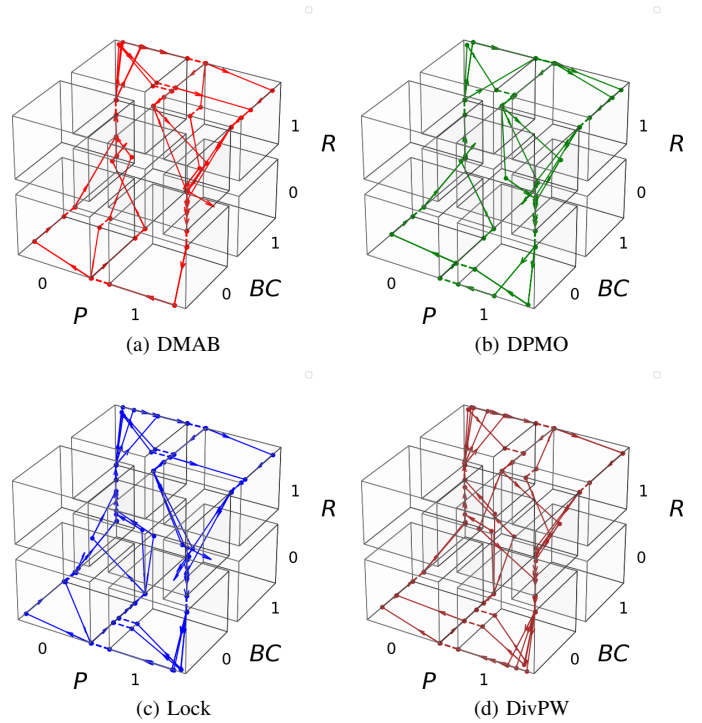


Fig. 4: Set of solutions extracted by the different planners in 3G.

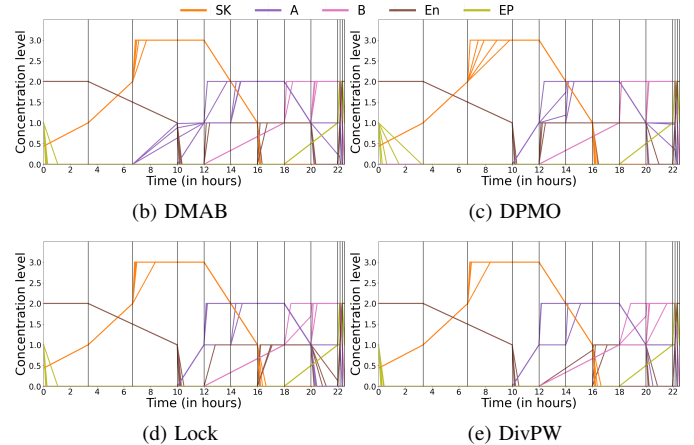


Fig. 5: Set of solutions extracted by the different planners in 5G.

cubes. In 5G (Figure 5), the graph represents the evolution of concentration (in the y-axis) as a function of the time spent (in the x-axis) for the different genes. Each vertical bar represents a discrete state change.

The different diverse planners proposed helped exhibit top-quality and diverse solutions, each consistent with BK. Between the different sets of plans displayed in 5G, DPMO's advantage in terms of diversity can be observed. In fact, the SK and EP celerities better cover the solutions space, in particular:

(i) between 0 and 4 hours, EP slopes (yellow curves) are more diverse, and (ii) between 6.30 and 10 hours, the same applies to SK slopes (orange curves).

V. CONCLUSION

This paper compares the DMAB baseline with different contributions to be applied with continuous MCTS. DPMO considers diverse planning in a multi-objective setting, Lock inhibits (locks) any new simulations for this plan, and DivPW biases the progressive widening formula. All of those planners are heuristics-based but showed that they help exhibit top-quality and diverse sets of plans. The experimental study has been conducted on a real-world problem in which the goal is to identify parameters of gene regulatory networks. Such identification is an ideal tool to help biologists develop hypotheses and facilitate the design of their experiments in the field of chronotherapy. This study is a continuation of [37] that tackles this task as a multimodal optimization problem.

Future works consider hybridising the DivPW and DPMO planners, while assessing their relevance on a more diverse set of benchmark functions. Another suggestion is to bridge the gap between optimization techniques employed for finding multiple solutions and online diverse planning with MCTS.

ACKNOWLEDGMENT

This work was supported by the French government through the France 2030 investment plan managed by the National Research Agency (ANR), as part of the Initiative of Excellence Université Côte d’Azur under reference number ANR-15-IDEX-01. The authors are grateful to the Université Côte d’Azur’s Center for High-Performance Computing (OPAL infrastructure) for providing resources and support.

REFERENCES

- [1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [2] R. Coulom, “Efficient selectivity and backup operators in monte-carlo tree search,” in *International conference on computers and games*, 2006, pp. 72–83.
- [3] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, “A survey of monte carlo tree search methods,” *IEEE Transactions on Computational Intelligence and AI in Games*, 2012.
- [4] M. Swiechowski, K. Godlewski, B. Sawicki, and J. Mańdziuk, “Monte carlo tree search: A review of recent modifications and applications,” *Artificial Intelligence Review*, 2023.
- [5] L. Kocsis and C. Szepesvári, “Bandit based monte-carlo planning,” in *European conference on machine learning*, 2006, pp. 282–293.
- [6] P. Auer, N. Cesa-Bianchi, and P. Fischer, “Finite-time analysis of the multiarmed bandit problem,” *Machine learning*, vol. 47, pp. 235–256, 2002.
- [7] B. Brüggmann, “Monte carlo go,” Technical report., Tech. Rep., 1993.
- [8] B. Bouzy and B. Helmstetter, “Monte-carlo go developments,” *Advances in Computer Games: Many Games, Many Challenges*, pp. 159–174, 2004.
- [9] S. Gelly and D. Silver, “Combining online and offline knowledge in uct,” in *Proceedings of the 24th ICML*, 2007.
- [10] —, “Monte-carlo tree search and rapid action value estimation in computer go,” 2011.
- [11] T. Cazenave, “Generalized rapid action value estimation,” in *24th IJCAI*, 2015, pp. 754–760.
- [12] R. Coulom, “Computing “elo ratings” of move patterns in the game of go,” *ICGA journal*, vol. 30, no. 4, pp. 198–208, 2007.
- [13] A. Couëtoux, J.-B. Hoock, N. Sokolovska, O. Teytaud, and N. Bonnard, “Continuous upper confidence trees,” in *Learning and Intelligent Optimization*, 2011, pp. 433–445.
- [14] G. M. J. Chaslot, M. H. Winands, H. J. v. d. Herik, J. W. Uiterwijk, and B. Bouzy, “Progressive strategies for monte-carlo tree search,” *New Mathematics and Natural Computation*, vol. 4, no. 03, pp. 343–357, 2008.
- [15] A. Couetoux, M. Milone, M. Brendel, H. Doghmen, M. Sebag, and O. Teytaud, “Continuous rapid action value estimates,” in *Asian conference on machine learning*, 2011, pp. 19–31.
- [16] R. Michelucci, D. Pallez, T. Cazenave, and J.-P. Comet, “Improving continuous Monte Carlo Tree Search for identifying parameters in hybrid gene regulatory networks,” in *Proceedings of the 18th International Conference on PPSN*, 2024.
- [17] J. Lee, W. Jeon, G.-H. Kim, and K.-E. Kim, “Monte-carlo tree search in continuous action spaces with value gradients,” in *AAAI*, 2020, pp. 4561–4568.
- [18] B. Kim, K. Lee, S. Lim, L. Kaelbling, and T. Lozano-Pérez, “Monte carlo tree search in continuous spaces using voronoi optimistic optimization with regret bounds,” in *AAAI*, 2020, pp. 9916–9924.
- [19] T. Yee, V. Lisỳ, M. H. Bowling, and S. Kambhampati, “Monte carlo tree search in continuous action spaces with execution uncertainty,” in *Proceedings of the 25th IJCAI*, 2016.
- [20] A. Ward, J. K. Liker, J. J. Cristiano, and S. D. K., “The second toyota paradox: How delaying decisions can make,” *Sloan Management Review*, vol. 36, no. 3, pp. 43–61, 1995.
- [21] D. Speck, R. Mattmüller, and B. Nebel, “Symbolic top-k planning,” in *AAAI*, vol. 34, no. 06, 2020, pp. 9967–9974.
- [22] M. Katz, S. Sohrabi, and O. Udrea, “Top-quality planning: Finding practically useful sets of best plans,” in *AAAI*, vol. 34, no. 06, 2020, pp. 9900–9907.
- [23] M. Katz and S. Sohrabi, “Reshaping diverse planning,” in *AAAI*, vol. 34, no. 06, 2020, pp. 9892–9899.
- [24] M. Katz, S. Sohrabi, and O. Udrea, “Bounding quality in diverse planning,” in *AAAI*, vol. 36, no. 9, 2022, pp. 9805–9812.
- [25] T. A. Nguyen, M. Do, A. E. Gerevini, I. Serina, B. Srivastava, and S. Kambhampati, “Generating diverse plans to handle unknown and partially known user preferences,” *Artificial Intelligence*, vol. 190, pp. 1–31, 2012.
- [26] L. Benke, T. Miller, M. Papisimeon, and N. Lipovetzky, “Diverse, top-k, and top-quality planning over simulators,” *26th ECAI*, 2023.
- [27] K. L. Myers and T. J. Lee, “Generating qualitatively different plans through metatheoretic biases,” in *AAAI/IAAI*, 1999, pp. 570–576.
- [28] B. Srivastava, T. A. Nguyen, A. Gerevini, S. Kambhampati, M. B. Do, and I. Serina, “Domain independent approaches for finding diverse plans,” in *IJCAI*, 2007, pp. 2016–2022.
- [29] A. Coman and H. Munoz-Avila, “Generating diverse plans using quantitative and qualitative plan distance metrics,” in *AAAI*, vol. 25, no. 1, 2011, pp. 946–951.
- [30] W. Wang and M. Sebag, “Multi-objective monte-carlo tree search,” in *Proceedings of the Asian Conference on Machine Learning*, 2012.
- [31] E. Zitzler and L. Thiele, “Multiobjective optimization using evolutionary algorithms—a comparative case study,” in *International conference on PPSN*, 1998, pp. 292–301.
- [32] J. Behaegel, J.-P. Comet, and F. Folschette, “Constraint identification using modified Hoare logic on hybrid models of gene networks,” in *Proceedings of the 24th Int. Symposium TIME*, 2017.
- [33] J. Behaegel, J.-P. Comet, G. Bernot, E. Cornillon, and F. Delaunay, “A hybrid model of cell cycle in mammals,” in *6th International Conference on Computational Systems-Biology and Bioinformatics*, 2015.
- [34] M. López-Ibáñez, J. Dubois-Lacoste, L. Pérez Cáceres, T. Stützle, and M. Birattari, “The irace package: Iterated racing for automatic algorithm configuration,” *Operations Research Perspectives*, pp. 43–58, 2016.
- [35] M. Kronfeld and A. Zell, “Towards scalability in niching methods,” in *IEEE CEC*, 2010.
- [36] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, “Optics: Ordering points to identify the clustering structure,” *ACM Sigmod record*, vol. 28, no. 2, pp. 49–60, 1999.
- [37] R. Michelucci, V. Callegari, J.-P. Comet, and D. Pallez, “Cellular genetic algorithms for identifying variables in hybrid gene regulatory networks,” in *Applications of Evolutionary Computation*, 2024.