Novelty Search: a Theoretical Perspective

Stephane Doncieux
Sorbonne Université, CNRS, Institut des Systèmes Intelligents et de Robotique, ISIR, F-75005 Paris, France
stephane.doncieux@sorbonne-universite.fr

Alban Laflaquière
AI Lab, SoftBank Robotics Europe, Paris, France
alaflaquiere@softbankrobotics.com

Alexandre Coninx
Sorbonne Université, CNRS, Institut des Systèmes Intelligents et de Robotique, ISIR, F-75005 Paris, France
alexandre.coninx@sorbonne-universite.fr

ABSTRACT

Novelty Search is an exploration algorithm driven by the novelty of a behavior. The same individual evaluated at different generations has different fitness values. The corresponding fitness landscape is thus constantly changing and if, at the scale of a single generation, the metaphor of a fitness landscape with peaks and valleys still holds, this is not the case anymore at the scale of the whole evolutionary process. How does this kind of algorithms behave? Is it possible to define a model that would help understand how it works? This understanding is critical to analyse existing Novelty Search variants and design new and potentially more efficient ones. We assert that Novelty Search asymptotically behaves like a *uniform random search process in the behavior space*. This is an interesting feature, as it is not possible to directly sample in this space: the algorithm has a direct access to the genotype space only, whose relationship to the behavior space is complex. We describe the model and check its consistency on a classical Novelty Search experiment. We also show that it sheds a new light on results of the literature and suggests future research work.

CCS CONCEPTS

• **Computing methodologies** → **Search methodologies**; **Evolutionary robotics**; *Neural networks*;

KEYWORDS

Novelty search; evolutionary robotics; theory; behavior space

ACM Reference Format:
Stephane Doncieux, Alban Laflaquière, and Alexandre Coninx. 2019. Novelty Search: a Theoretical Perspective. In *Genetic and Evolutionary Computation Conference (GECCO '19), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3321707.3321752

1 INTRODUCTION

Evolutionary algorithms rely on a selection process driven by a fitness function that measures to what extent an individual satisfies a given goal. It has been shown that this goal-oriented fitness function may be misleading and can actually be replaced by a goal-independent selective pressure: a pressure towards *novel* behaviors [9]. Surprisingly, although the goal to achieve is not taken into account *at all* in the search process, the resulting Novelty Search (NS) process appears to be at least as efficient, if not more efficient, than a goal-oriented search in a certain number of domains like maze navigation and biped locomotion [9], swarms of robots [5], or plastic neural networks design [20].

Evolutionary processes driven by a constant goal have a dynamic that can be understood in terms of movements in a fitness landscape [27]: the evolutionary process tends to drive the individuals towards peaks of fitness. In NS, the fitness landscape changes from one generation to another, as an individual that is considered as new in one generation is likely to be much less novel in the next. *What is the dynamic of a NS process? How can it be modeled?* The lack of answer to these questions has practical consequences. When trying to implement such an algorithm, many important questions remain unanswered: how to select the behavior descriptors? What are the most critical properties or parameters of NS? Several empirical studies have explored the impact of a range of algorithmic choices [4, 7], but there is still no theoretical model that integrates all of their results to propose a unifying perspective of the approach. Such a model would be interesting to evaluate the consequences of design choices, and propose more efficient algorithms. Furthermore, NS is also at the basis of Quality-Diversity algorithms [2, 18]. These algorithms aim at finding a diverse set of efficient solutions, instead of a single best performing solution. Their outcome is a container that is filled during the search process by novelty-like principles. Better understanding how NS works can thus also open the way to better Quality-Diversity algorithms.

Novelty Search algorithms and their variants are expected to lead to a good exploration of the behavior space [2, 18], and even to explore it uniformly [4]. In this paper, we introduce a simplified and theoretical model of novelty-driven algorithms based on the assumption that NS leads to an *indirect and uniform random search process in the behavior space*. A model is built on the basis of this assumption, and its properties are analyzed and discussed and an extension is proposed. These theoretical models are then compared to experimental results, and their relevance is discussed in light of the literature.

2 NOVELTY SEARCH

We take inspiration from the frequently used Reinforcement Learning framework to define the notations used throughout the paper. In relation with the illustration of Figure 1, let's define the tuple $< S, A, m, \pi >$ as follows:

- S is a state space, gathering all states the agent can be in. A state is supposed to contain the information required by the agent to decide what action to perform,

GECCO '19, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6111-8/19/07…$15.00
https://doi.org/10.1145/3321707.3321752

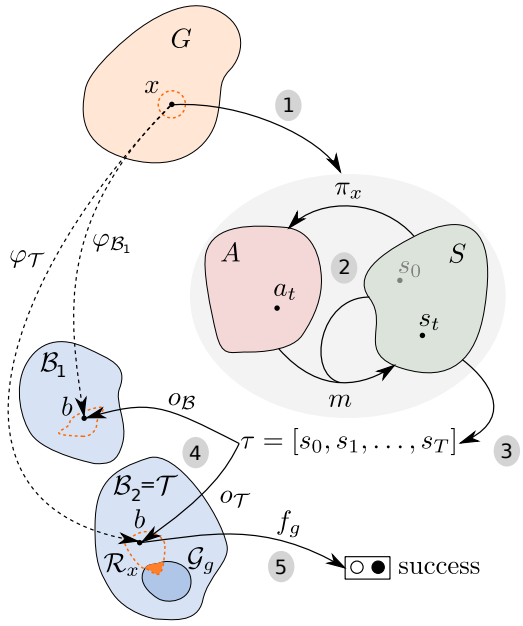

Figure 1: Overview of the different spaces and functions related to Novelty Search. The Genotype space G describes a policy π (1) that determines the actions in A to apply in a given state $s \in S$ (2). The sequence of states describes the robot behavior (3) that is projected to lower dimension behavior spaces $\mathcal{B}_1$ or $\mathcal{B}_2$ (4). Note that $\mathcal{B}_2$ can also be called $\mathcal{T}$ as it defines a task. In this case, $\mathcal{G}_g$ describes the goal to be reached (5) and $\mathcal{R}_x$ the area of the task space $\mathcal{T}$ that is within a single mutation range from the genotype x.

- A is an action space, gathering all the actions the agent can take,
- m is a transition function that describes how the agent's state changes as a result of its action:

$$m : S \times A \to S,$$

- π is a policy, a function that associates an action $a \in A$ to each state $s \in S$ of the agent:

$$\pi : S \to A.$$

As a first approximation, and to simplify notations, we consider that m and π are deterministic functions. Likewise, we consider that the agent has a direct access to its state (or equivalently that its sensors provide a state). In practice, it only has access to the observations of its sensors. In the Reinforcement Learning theoretical framework, the state is expected to have some properties, one of the most important being the Markovian property, the fact that the states contains all information to make a decision, i.e. that we don't need to take into account the past or that there is no noise. As we thought it would not have a strong impact on our model, at least at the level of analysis presented in this article, we have neglected this aspect to simplify the notations.
Note that, for reasons that will appear clearer below, we omit the usual reward function that associates a reward to each state.

In order to take into account the specificities of the NS approach, we define additional spaces and functions:

- G is a genotype space, a space of parameters defining the policy π, and that NS algorithms directly explore. A policy parametrized by $x \in G$ will be denoted π_x,
- $\mathcal{B}$ is a behavior space used to describe the agent's behavior,
- $o_{\mathcal{B}}$ is an *observer function*, which associates a behavior descriptor $b \in \mathcal{B}$ to a trajectory of states $\tau = \{s_0, s_1, \ldots, s_T\}$ of length T:

$$o_{\mathcal{B}} : S^T \to \mathcal{B}.$$

  In most cases, the characterization b of the agent's behavior is more compact than the complete trajectory τ, as the latter contains a lot of superfluous information with respect to the final task to solve,
- f_g is a *goal function*, which defines a particular goal to reach by associating a behavior $b \in \mathcal{B}$ with a boolean indicating if it reaches the goal or not:

$$f_g : \mathcal{B} \to \mathbb{B} = \{\text{True}, \text{False}\}.$$

  When a goal function f_g is associated with a behavior space $\mathcal{B}$, we will use the Machine Learning notion of *task* and refer to this space as a *task space* and denote it $\mathcal{T}$. The concept of goal function is very similar to the one of reward function usually used in Reinforcement Learning, except that it is defined in a behavior space instead of a state space. Note that this definition of a goal function is not suitable for non-boolean goals, like for instance when the task is to maximize the speed of an agent. Such situations can be covered by this definition by discretizing the continuous goal and associating a separate goal to each of the corresponding bins. This is for instance done in the MAP-Elites algorithm [15],
- $\mathcal{G}_g$ is a goal space. It is the support set of f_g in the task space $\mathcal{T}$, such that:

$$\forall b \in \mathcal{G}_g \subseteq \mathcal{T}, f_g(b) = \text{True}.$$

The different spaces and functions we just defined are represented in Figure 1. It is possible to abstract the whole process leading to the generation of a behavior by defining a mapping $\varphi_{\mathcal{B}}$ between a genotype space G and a behavior space $\mathcal{B}$. It corresponds to a combination of π_x, m, and $o_{\mathcal{B}}$ such that:

$$\varphi_{\mathcal{B}}(x) = o_{\mathcal{B}}(s_0, s_1, \ldots, s_T), \text{ with } s_{t+1} = m(s_t, \pi_x(s_t)).$$

We call this overarching function a *behavior function* [7].

The motivation behind the proposed goal function definition is the hypothesis behind NS: a fitness gradient may not always be available and/or might be misleading, and it is therefore more appropriate not to follow it. The function f_g is boolean, and thus does not create a gradient to follow. Novelty Search replaces the search for a greater fitness by a search for novel individuals. Its selection process is driven by a novelty measure derived from the distance between a new individual and its nearest neighbors in a behavior space [9]:

$$\rho(x) = \frac{1}{k} \sum_{i=1}^{k} \text{dist}\big(\varphi_{\mathcal{B}}(x), \varphi_{\mathcal{B}}(\mu_i)\big), \tag{1}$$

where $\{\mu_1, \ldots, \mu_k\}$ is the set of the k closest individuals to x in the behavior space, and $dist(.,.)$ is a distance in the behavior space.

In general, the set of individuals used to measure the novelty of new solutions corresponds to the current population, plus an archive of previous individuals, and the Euclidean distance is used to compare behaviors. Several strategies exist to manage the archive of past solutions [4]. It can contain either the individuals whose novelty was above a threshold when first evaluated [9], the most novel individuals at each generation [14], randomly chosen individuals [8], or even none at all [16]. Solving a particular task consists in discovering at least one genotype x for which $f_g(\varphi_{\mathcal{T}}(x)) = \text{True}$. As already mentioned, no reward gradient is used to guide the search. A genotype x is either a solution or not, and no intermediate value indicates how far an individual is from a solution.

Note that, in this framework, the goal is defined in a behavior space (making it a task space $\mathcal{T}$). However, this space cannot be directly explored. The search process can only sample the genotype space G, and the mapping $\varphi_{\mathcal{T}}$ between G and $\mathcal{T}$ is a priori unknown and non-trivial. For instance similar genotypes can lead to very different behaviors, and inversely. Despite this difficulty, NS is designed to explore the space of behaviors as efficiently as possible, making it easier to find a solution.

In this paper, we propose to formally analyze how NS approaches explore the task space and find solutions. We hope that a better understanding of its dynamic will guide the future design of more efficient novelty-based approaches. It should be noted that NS algorithms have to be coupled with generation algorithms that produce new individuals to evaluate. This generation process is often done via neuroevolution methods that are variants of NEAT [22]. Although this generation component is taken into account in the experiments of Sections 3.2 and 4.2, the model proposed in this paper focuses on the selective pressure induced by NS algorithms.

3 UNIFORM SAMPLING MODEL

3.1 Model description

We've seen in Section 2, that NS is designed to explore a behavior space efficiently. As a first approximation, we will consider that NS, globally and after a transient phase, performs an ideal random and uniform search in the behavior space $\mathcal{T}$ in which the task is defined. This first model does not aim at depreciating the transient phase, but at studying the steady phase. The transient phase is clearly important in NS and will be considered in the model proposed in the next section.

When the goal function is boolean, gradient-based approaches cannot be used, and, if no prior is available on the task, approaches that model the goal, like Bayesian optimization, are also inefficient. The best search strategy thus consists in a systematic exploration of the search space. Without a priori knowledge about the task space, a random and uniform search in the task space is thus a good option, if not the best. However this is not a trivial problem in practice, as the search algorithm can sample $\mathcal{T}$ only indirectly, via a sampling of G. In order to derive an upper bound on the efficiency of NS, we nonetheless assume here that the algorithm is capable of such an ideal uniform sampling.

Accordingly, if $b \in \mathcal{T}$ is the behavior of an individual generated by a uniform random search process, the probability p_1 that $b \in \mathcal{G}_g \subset \mathcal{T}$ is a solution is:

$$p_1 = \frac{|\mathcal{G}_g|}{|\mathcal{T}|}, \tag{2}$$

where $|.|$ denotes the cardinality of a set. Likewise, the probability p_n that at least one solution has been found after n samples is:

$$p_n = 1 - (1 - p_1)^n = 1 - \bar{p_1}^n,$$

which corresponds to the complement of not finding a solution n times. This defines the dynamics of success of an ideal uniform search, and should thus act as a statistical upper bound on the performance of NS algorithms.

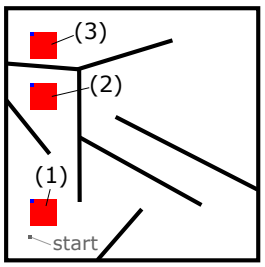

Figure 2: Experimental maze in which (1) is the start region, (2) is the dead-end region, and (3) is the exit region. The red squares correspond to large regions representing 1/100 of the total maze area. The blue squares correspond to small regions representing 1/10000 of the total area.

In line with the uniform sampling assumption, and with the goal to test it with a dedicated experiment, we formulate different hypotheses about the exploration produced by NS algorithms:

Hypothesis 1. *The sampling produced by NS covers the whole reachable behavior space.*

Hypothesis 2. *The sampling performed by NS tends towards a uniform distribution in the reachable behavior space.*

Finally, it is possible to set up NS such that it explores a behavior space $\mathcal{B}$ that differs from the task space $\mathcal{T}$. If the behavior descriptors of $\mathcal{B}$ are not related to the task to solve, we say that $\mathcal{B}$ is *unaligned* with the task. In such a context, NS revealed to perform poorly on difficult mazes [18]. The interpretation of these results with the proposed model is straightforward: if a uniform exploration in $\mathcal{B}$ does not imply a uniform exploration in $\mathcal{T}$ (or at least of a subset of $\mathcal{T}$ containing $\mathcal{G}_g$), the search will likely fail.

Hypothesis 3. *Performing NS on a behavior space $\mathcal{B}$ unaligned with the task space $\mathcal{T}$ does not necessarily result in a uniform exploration of $\mathcal{T}$.*

We will test these hypotheses and the consistency of the proposed model by running some NS experiments.

3.2 Experiments

We propose to test the model on the typical maze experiment proposed in the original NS article [9]. As illustrated in Figure 2, a two-wheeled robot has to navigate a room obstructed with walls until it reaches an exit. The agent is equipped with 3 distance sensors. It is controlled by a neural network with 3 inputs, plus a constant input,

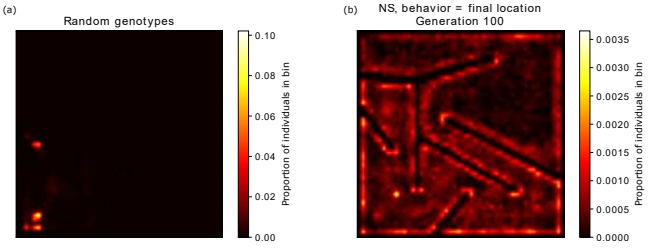

Figure 3: Solutions generated by (a) a random search of 10000 individuals in the genotype space and by (b) 100 generations of 100 individuals via novelty search.

and 2 outputs corresponding to the speed of the two wheels. A direct encoding of the network based on NEAT [22] is used, without crossover, as described in [16]. With the "start simple" feature of NEAT, it means 8 continuous parameters to explore at the beginning of the search, this number growing or shrinking along a run, and for each individual, depending on structural mutations (adding or removing neurons or connections). Following the NS approach, the selective pressure is the novelty of an individual behavior. The characterization of a behavior consists in taking the 2-dimensional position of the robot at the end of an episode of $T = 2000$ iterations. Note that the size of the behavior space is smaller than the one of the genotype space, but this does not appear to be a critical property, as it has been shown that the size of the behavior characterization had a limited impact [9]. The size of the population at each generation is set to 100, the size of the archive is unbounded, and, as suggested in [4, 9], the number of neighbors to estimate novelty is $k = 15$. After each generation is evaluated, the most novel individual is added to the archive. The source code of the experiment is available at https://github.com/doncieux/navigation.git, in the ns_theory branch.

With respect to the proposed formalism, the state space S is the space of wall distances and robot positions. The wall distance is used by the policy and the observer function consists in extracting the final robot position. The action space A is the space of motor actions.

Figure 3 shows the solutions explored by a random search in the genotype space and by the considered NS algorithm. The random search results in a poor exploration of the task space (see Figure 3(a)), whereas NS appears to cover the whole reachable task space during the first 100 generations (see Figure 3(b)). This result is compatible with hypothesis 1.

Figure 4 shows the evolution of the coverage of the behavior space by the NS algorithm. During the first generations, this coverage is not uniform (see Figure 4), but it rapidly tends towards a more uniform distribution after several generations (see Figure 4). This result is compatible with hypothesis 2.

Note however that the coverage produced by NS exhibits a certain dynamic. The probability to generate an individual in a given region of the task space is not constant throughout the generations, like it would be the case with an ideal uniform sampling. We propose to study this property by considering different regions in the task space, and estimate the probability to reach them throughout the exploration. We have chosen 3 different regions: one near the robot's starting position (1), one in the dead-end (2) and the last one near the exit (3) (see Figure 2). Two different region sizes are considered:

- large: the large regions are squares of size 0.1×0.1 relative to the size of the environment. Due to the presence of obstacles (walls), the area reachable by the agent is equal to 68.9% of the total area of the environment. The theoretical value of p_1 is thus equal to: $0.01/(1 \times 0.689) = 14.51 \times 10^{-2}$.
- small: the small regions are squares of size 0.01×0.01 relative to the size of the environment. They correspond to a theoretical probability p_1 equal to: 14.51×10^{-4}.

In the experiment, p_1 is empirically estimated by measuring the total number of individuals whose behavior is in a given region, divided by the total number of individuals generated so far. Figure 5 shows how this empirical probability evolves against the number of generations. The large region (1), next to the starting position of the robot, has a probably to be reached above 0.06 at the beginning of the exploration. This is not surprising as many randomly generated individuals may not move a lot. The large dead-end region (2) is reached after only a few generations, whereas the large exit region (3) starts to be reached, i.e. $p_1 > 0$, only after 49 generations (median). A similar evolution of the probability can be observed for the small regions. This behavior is not surprising. It corresponds to a bootstrap phase of NS, that needs to discover the genotypes that will explore regions that the majority of random networks do not explore. After this initial phase, the estimation of p_1 converges to similar values for the three areas. Furthermore, this value is close to the theoretical value. This result is thus conform to the asymptotic uniformity hypothesis (hypothesis 2). Robots may end up blocked against a wall. It results in a bias with more solutions against walls, as can be seen in Figure 4. This may explain why the estimated value of p_1 is below the theoretical one.

So far, we have considered experiments in which the behavior space explored by NS is the task space. In order to test hypothesis 3, we run the same experiment, except that the behavior descriptor corresponds this time to the orientation of the robot at the end of the episode [18]. As shown in Figure 6, the sampling is uniform in the "orientation" behavior space, but not in the end position space. As a consequence, the empirical value of p_1 converges to different values for the three regions. This result is compatible with hypothesis 3.

3.3 Interpretation and consequences

In this model, the probability that at least a solution was found after n samples solely depends on $p_1 = \frac{|\mathcal{G}_g|}{|\mathcal{T}|}$. It has several consequences.

The larger $\mathcal{G}_g$, the higher the probability to find a solution. In the maze experiments, for instance, the upper bound of the probability to find a solution depends on the relative size of the exit area. This parameter is not usually described in the literature, whereas it has a significant impact on the results according to our model and as shown on Figure 5. Likewise, it may also explain why *novelty search performs poorly in the case of an open environment* [9]: if the environment is open, $\mathcal{T}$ is unbounded. If $\mathcal{G}_g$ is bounded, p_1 then tends to 0.

The number of behavior descriptors is inconsequential if it does not impact p_1. If increasing the dimension of the behavior space does

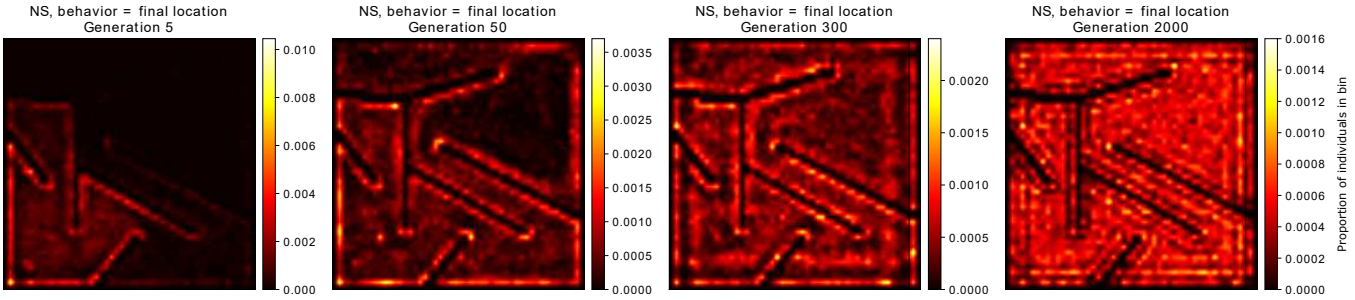

Figure 4: Solutions generated by novelty search experiments at generations 5, 50, 300, and 2000. They are plotted as a heatmap. The environment is divided in 50x50 bins and the color represents the number of individuals in the corresponding bin.

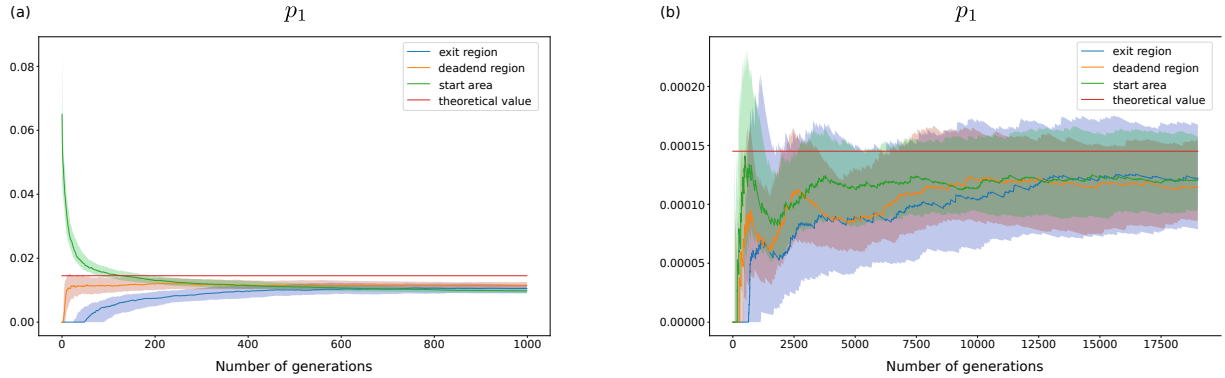

Figure 5: Evolution of the probability p_1 to reach (a) large regions and (b) small regions. The figures display the median and the 25 and 75 percentile curves out of 150 runs.

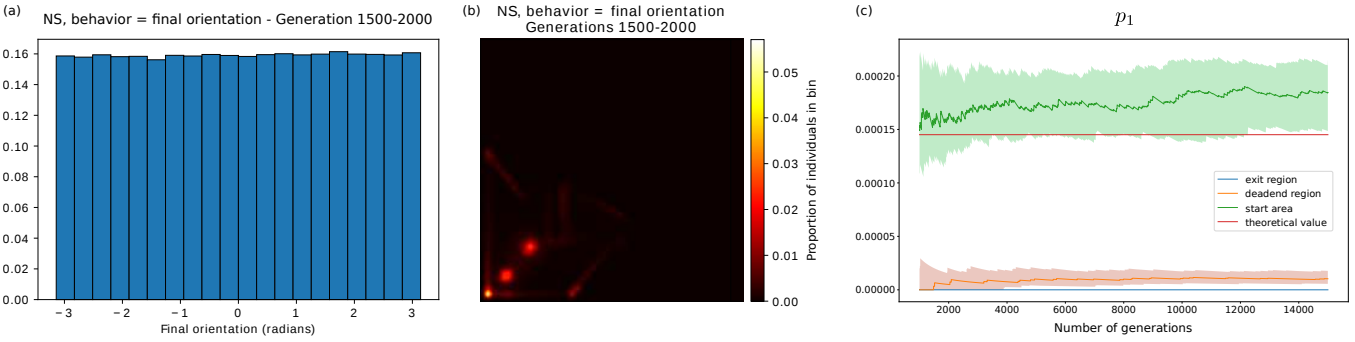

Figure 6: Experiment with NS when the behavior space corresponds to the final orientation of the robot. (a) Distribution of the final orientation of the robot over generations 1500 to 2000. (b) Distribution of the robot final positions in the (unaligned) task space. (c) Evolution of p_1 for the small regions.

not modify p_1, then it has no impact on the upper bound of the probability of convergence. This is what has been observed in [9]. In the original maze experiments, the task space corresponds to the space of robot positions at the end of an episode. If some intermediate points of the trajectory are added as behavior descriptors, p_1 remains unchanged as these new dimensions have no impact on the success or failure of the individual: if $b_T = (x_T, y_T)$ is in the goal space $\mathcal{G}_g$, any $b' = (x_t, y_t, x_T, y_T)$, with $t < T$, is also a solution, regardless of the added descriptors (x_t, y_t), and thus $p'_1 = p_1$.

4 MODEL INCLUDING REACHABILITY

The uniform sampling model proposed in Section 3 is an *ideal* model of how NS behaves, just as a search process climbing fitness peaks is an ideal model of how goal-oriented evolutionary algorithms behave. A typical evolutionary algorithm does not systematically follow the fitness gradient, but balances exploration and exploitation [1, 3]. The result is that, although a goal-oriented evolutionary algorithm can be seen as a fitness peak climber, new genotypes do not all correspond to an increase in fitness. This is the same for the

uniform sampling model of NS: although we assumed the algorithm leads to a uniform sampling of the behavior space, a significant part of the generated individuals may correspond to already sampled behaviors, in particular at the beginning of a run. Offspring may, for instance, have the same behavior as the one of their parents, or a significant number of them may have a similar behavior (not moving at all, for instance, or bumping into a wall). More generally, the generation of new individuals might be skewed such that only a subpart of the behavior space can be sampled at a given generation. It corresponds to a loss of search efficiency. In this section, we refine our model to capture this phenomenon.

The proposed extension relies on the concept of evolvability. This notion is essential to evolution, but its precise definition is debated [17]. For Wagner and Altenberg, it is "the ability of random variations to sometimes produce improvement" [25]. For Kirschner and Gerhart, it is "the capacity to generate heritable, selectable phenotypic variation" [6]. They further add that "this capacity may have two components: (i) to reduce the potential lethality of mutations and (ii) to reduce the number of mutations needed to produce phenotypically novel traits". In this paper, we consider that evolvability characterizes the fact that the generation of new individuals might only cover a subpart of the behavior space explored. It represents only one of the facets of evolvability. To avoid any ambiguities, we will thus call it with a different name: the *reachability*.

4.1 Model description

Let us call $\mathcal{R}_x$ the subpart of the behavior space reachable by applying random mutations on the genotype x (see Figure 1). It corresponds to the evolvability of genotype x according to Kirschner and Gerhart's definition and follows the one used in [12]. Contrary to the uniform model, the probability to generate a solution in this second model depends on the genotype used to generate the new sample:

$$p_{1,x} = \frac{|\mathcal{G}_g \bigcap \mathcal{R}_x|}{|\mathcal{R}_x|} = \frac{\epsilon_{x,g}|\mathcal{G}_g|}{\alpha_x|\mathcal{T}|} = \gamma_{x,g} \cdot \frac{|\mathcal{G}_g|}{|\mathcal{T}|} = \gamma_{x,g} . p_1, \quad (3)$$

$\epsilon_{x,g}$ and α_x are respectively the proportions of $\mathcal{G}_g$ and $\mathcal{T}$ within reach from x, and $\gamma_{x,g} = \frac{\epsilon_{x,g}}{\alpha_x}$ is the *reachability of the goal g from genotype x*.

The reachability is individual-based and does not allow to make predictions if the corresponding genotypes are unknown, but reachability can also be seen at the population level [12, 13, 26]. To this end, we assume that reachability is a population-based property that depends on the number of individuals sampled so far, and not on the particular genotype x a new individual is generated from, i.e.: $\gamma_{x,g} = \gamma_{n,g}$. This assumption simplifies Equation 3 and provides a bound on the probability of success. We can see $\gamma_{n,g}$ as the *reachability of the goal g after n individuals have been sampled.*

The probability of success $p_n^{(g)}$ of the model with evolvability is:

$$p_n^{(g)} = 1 - \prod_{k=1}^{n} (1 - \gamma_{k,g} p_1), \quad (4)$$

which corresponds to the complement of the probability of not finding a solution n times.

$\gamma_{k,g}$ may be higher than 1. Actually, if $\mathcal{R}_x = \mathcal{G}_g$, $p_{1,x} = 1$ and thus $\gamma_{x,g} = \gamma_{n,g} = \frac{1}{p_1}$. This situation happens if the sampling is not uniform in the behavior space. This has been invalidated by the previous section, at least in the proposed experiment and in the steady phase. The theoretical models we will consider to fit the experimental data will thus make the assumption that the sampling is uniform and not biased, i.e. that p_1 is an upper bound of $p_{1,x} = p_{1,n}$ and thus that $\gamma_{n,g} \leq 1$. The corresponding theoretical evolution of p_n with reachability is displayed in Figure 7 for two different scenarios: $\gamma_{n,g} = \gamma_g = constant$, and $\gamma_{n,g} = 1 - exp(\frac{-n}{k_g})$. This second model corresponds to a reachability that increases over the number of samples to asymptotically converge to 1. In both cases, the impact of the reachability is a delay in the increase of p_n. The main difference between the two models is the slope, that is steeper for the model with an increasing reachability.

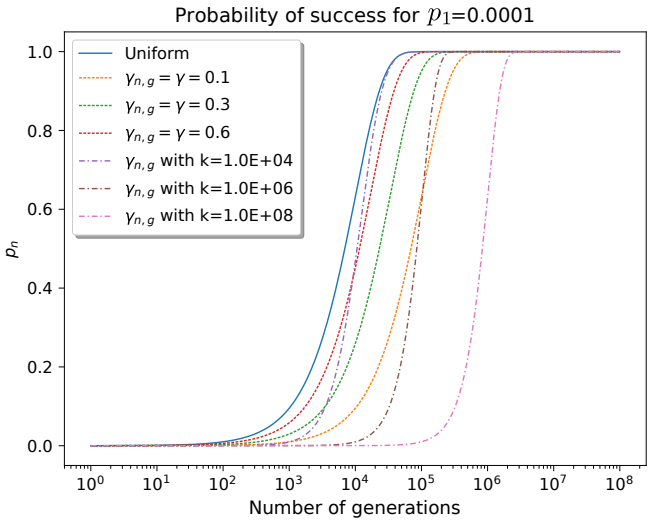

Figure 7: Probability to generate a solution in a region w.r.t. the number of samples n in the model, taking into account the reachability with $p_1 = 0.0001$ and γ either constant or increasing with the number of samples: $\gamma_{n,g} = \gamma_n = 1 - exp(\frac{-n}{k})$. There are 100 samples per generation.

This extended model allows us to formulate the following hypothesis:

Hypothesis 4. *The increase of p_n is delayed with respect to the theoretical value derived from the uniform sampling model, and this delay depends on the goal.*

Following the literature [12], we can also make the following hypothesis:

Hypothesis 5. *The reachability $\gamma_{n,g}$ increases with the number of sampled individuals.*

4.2 Experiments

The following experiments rely on the same maze navigation setup as in Section 3.2. Figure 8 shows, for regions (1) and (3), the value of p_n estimated over 150 runs, together with curves derived from the theoretical model of p_n with reachability. The theoretical curves are optimized for both models of $\gamma_{n,g}$. The values of γ_g and k_g are estimated by dichotomy to make the theoretical curve fit with the

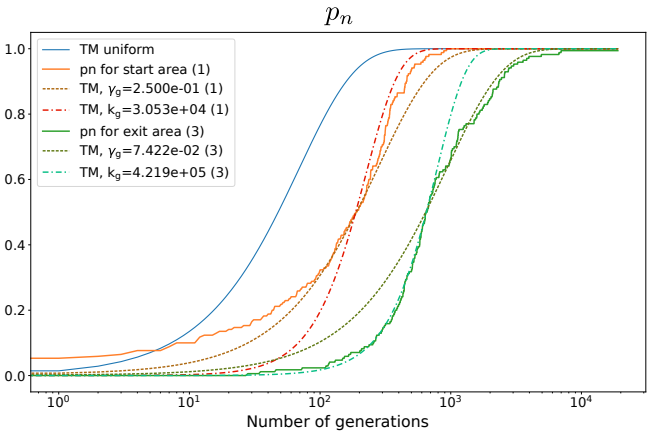

Figure 8: Estimation of p_n on the basis of 150 runs. The values of the theoretical models (TM) corresponds to $p_1 = 0.0001451$ and either a constant reachability ($\gamma_{n,g} = \gamma_g$) or an increasing one ($\gamma_{n,g} = 1 - exp(\frac{-n}{k_g})$, in this case, k_g value is indicated). There are 100 samples per generation.

empirical one for $p_n = 0.5$. The results of region (2) have been omitted to improve the readability of the figure.

The variations of p_n for both regions are similar to the one of the theoretical uniform sampling model, but with a delay that differs between the two goals. The delay is larger for the exit region (3), that is more difficult to reach. The estimated value of p_n for region (1) starts with an initial value above the theoretical one. It shows that, at this moment, the sampling is biased towards that area. It is not surprising as many randomly generated individuals remain close to the starting position (Figure 3(a)). Except at this moment, hypothesis 4 thus seems to be valid in this experiment.

The increase of reachability during exploration has been experimentally observed, and is thought to be due to the divergent search property of NS [12]. Yet, according to Figure 8, the hypothesis 5 is not fully validated by our model, that shows a more complex picture. The fit with the two proposed reachability models (constant vs increasing value) suggests a reachability whose behavior changes along the generations. For the start region (1), the fit is better with a constant reachability before the inflexion point ($p_n = 0.5$), and better with an increasing reachability after the inflexion point. On the contrary, for the exit region (3), the pattern is reversed, and the fit is better with an increasing reachability first, and a constant reachability after the inflexion point. These results suggest that the evolution of reachability requires a more refined model.

4.3 Interpretation and consequences

A low reachability slows down the exploration process and creates a significant delay before reaching the goal space. The reachability γ_g better fitting the empirical data is 0.25 for region (1) and 0.074 for region (3). $p_n = 0.5$ is reached at generation 48 for the theoretical model and empirically at generation 187 for the region (1) and 645 for the region (3) (Figure 8). Actually, when γ is constant, it can be shown that the number of samples required to reach a probability of success $p_n = p_s$ is: $\frac{\ln(1-p_s)}{\ln(1-\gamma p_1)}$. It further confirms that this criterion needs to be maximized.

Typical measures of evolvability estimate the diversity of mutated solutions [11, 21, 23]. It roughly corresponds to the average evolution speed at a given generation, whereas p_n and the reachability can be used to estimate the length of the path to reach a given goal space when starting from randomly generated solutions.

A fundamental feature of the reachability and p_n is that they are specific to a given goal space and thus highlight that *each region of the task space may have a different reachability*. The diversity of mutated solutions in the genotype space does not take into account the mapping between the genotype and the goal spaces and can thus hardly capture these differences. Variations in the reachability of different regions is an estimation of the difficulty to reach some parts of the task space. *The reachability estimates two features of a particular NS algorithm*: its speed to cover the whole task space when we consider its average value, and its biases when we look at its inter-region variations. Both pieces of information are important and can lead to more refined comparisons of alternative NS algorithms, including the selection algorithm and the underlying neuroevolution part.

In Section 3, it was hypothesized that NS performs, asymptotically, a uniform search in the behavior space. The differences of reachability between the regions highlight the differences in the transient phase, not the differences in the following phase: once a region has been reached, it tends to be sampled as frequently as the others, as shown by Figure 5. *The reachability then characterizes the dynamics of the search process and how fast it tends to be uniform.* As shown on Figure 8, the reachability also has its own dynamics that needs to be further studied.

5 DISCUSSION

The main motivation for NS is to enhance exploration. It can be for two reasons: (1) reaching a particular goal space as rapidly or as frequently as possible, this is what was emphasized in the first works on novelty search [9], or (2) generating a set of solutions that is as diverse as possible while not targeting a particular goal space, as in QD algorithms [2, 18].

To reach a particular goal space more rapidly, a first possibility suggested by the proposed model is, of course, to increase the probability to generate a solution $p_1^{(g)} = \frac{|\mathcal{G}_g|}{|\mathcal{T}|}$, which is problem-dependent and completely independent from the search process. There are at least two different ways to influence it: (1) change the genotype space G to include a statistical bias in favor of the goal space $\mathcal{G}_g$, or (2) change the task space $\mathcal{T}$ to a new task space $\mathcal{T}'$ such that $\frac{|\mathcal{G}'_g|}{|\mathcal{T}'|} > \frac{|\mathcal{G}_g|}{|\mathcal{T}|}$. The efficiency of the task space sampling depends on the reachability and designing a new task space in which p_1 is higher, may result in a lower reachability. Let us consider a task space $\mathcal{T}' = \{b_1, b_2\}$ in which the behavior function $\varphi_{\mathcal{T}'}$ projects the genotypes with right behaviors on b_1 and those with the wrong behaviors on b_2, i.e. $f_g(b_1) = True$ and $f_g(b_2) = False$. It results in $p_1 = 0.5$, but the reachability would be very low as most individuals are likely to get a behavior descriptor equal to b_2. Building a new task space then requires to find an appropriate tradeoff between p_1 and $\gamma_{n,g}$.

A second possibility suggested by the model is to increase the reachability $\gamma_{n,g}$. The reachability depends on the genotype [19, 24], but also on the evolutionary process [12, 13, 26]. Likewise, the behavior function has a critical impact [7]. All the cited studies have been performed in different contexts. The proposed theoretical model may help to combine them in a common framework.

QD-algorithms based on novelty search, like Novelty Search with Local Competition [10], consider the archive as the outcome of the algorithm [2, 18]. Their goal is to generate a set of solutions that is both efficient and diverse. The uniformity of the search promoted by the novelty criterion is thus critical for such algorithms as it will better cover the set of reachable behaviors and promote better local competitions. For these algorithms, it is then important to look for *uniform reachabilities*, i.e. reachabilities that do not depend on the goal: $\gamma_{n,g} = \gamma_n$.

MAP-Elites is a QD algorithm based on a grid [15]. It has the interesting feature to provide insights on the behavior space[1] and it is thus used as an analysis tool [15, 18, 23]. The proposed model could help analyse its efficiency with goal spaces corresponding to MAP-Elites grid cells.

6 CONCLUSION

We have introduced a simple model to analyze and interpret Novelty Search algorithms. The model is based on the hypothesis that NS algorithms perform an indirect, uniform random search in the behavior space. This is an interesting feature as this space cannot be directly sampled. Experiments done on the deceptive maze of Lehman and Stanley are consistent with the model. We have extended the model to include evolvability as a reachability in the task space and also checked the consistency of this extension with the experimental data.

This model is expected to help people willing to use Novelty Search to understand how it works and what it aims at. It also sets theoretical boundaries to the expected performance of these algorithms.

ACKNOWLEDGEMENT

This work was supported by the DREAM project[2] through the European Union's Horizon 2020 research and innovation program under grant agreement No 640891.

REFERENCES

[1] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. 2013. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR)* 45, 3 (2013), 35.
[2] Antoine Cully and Yiannis Demiris. 2018. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation* 22, 2 (2018), 245–259.

[1] It has thus been called an illumination algorithm.
[2] http://www.robotsthatdream.eu

[3] Agoston E Eiben and Cornelis A Schippers. 1998. On evolutionary exploration and exploitation. *Fundamenta Informaticae* 35, 1-4 (1998), 35–50.
[4] Jorge Gomes, Pedro Mariano, and Anders Lyhne Christensen. 2015. Devising effective novelty search algorithms: A comprehensive empirical study. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, 943–950.
[5] Jorge Gomes, Paulo Urbano, and Anders Lyhne Christensen. 2013. Evolution of swarm robotics systems with novelty search. *Swarm Intelligence* 7, 2-3 (2013), 115–144.
[6] Marc Kirschner and John Gerhart. 1998. Evolvability. *Proceedings of the National Academy of Sciences* 95, 15 (1998), 8420–8427.
[7] Steijn Kistemaker and Shimon Whiteson. 2011. Critical factors in the performance of novelty search. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 965–972.
[8] Joel Lehman and Kenneth O Stanley. 2010. Efficiently evolving programs through the search for novelty. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM, 837–844.
[9] Joel Lehman and Kenneth O Stanley. 2011. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation* 19, 2 (2011), 189–223.
[10] Joel Lehman and Kenneth O Stanley. 2011. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 211–218.
[11] Joel Lehman and Kenneth O Stanley. 2011. Improving evolvability through novelty search and self-adaptation.. In *IEEE Congress on Evolutionary Computation*. 2693–2700.
[12] Joel Lehman and Kenneth O Stanley. 2013. Evolvability is inevitable: Increasing evolvability without the pressure to adapt. *PloS one* 8, 4 (2013), e62186.
[13] Joel Lehman, Bryan Wilder, and Kenneth O Stanley. 2016. On the critical role of divergent selection in evolvability. *Frontiers in Robotics and AI* 3 (2016), 45.
[14] Antonios Liapis, Georgios N Yannakakis, and Julian Togelius. 2015. Constrained novelty search: A study on game content generation. *Evolutionary computation* 23, 1 (2015), 101–129.
[15] Jean-Baptiste Mouret and Jeff Clune. 2015. Illuminating search spaces by mapping elites. *CoRR* abs/1504.04909 (2015). arXiv:1504.04909 http://arxiv.org/abs/1504.04909
[16] Jean-Baptiste Mouret and Stephane Doncieux. 2012. Encouraging behavioral diversity in evolutionary robotics: An empirical study. *Evolutionary computation* 20, 1 (2012), 91–133.
[17] Massimo Pigliucci. 2008. Is evolvability evolvable? *Nature Reviews Genetics* 9, 1 (2008), 75.
[18] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. 2016. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI* 3 (2016), 40.
[19] Joseph Reisinger and Risto Miikkulainen. 2007. Acquiring evolvability through adaptive representations. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, 1045–1052.
[20] Sebastian Risi, Charles E Hughes, and Kenneth O Stanley. 2010. Evolving plastic neural networks with novelty search. *Adaptive Behavior* 18, 6 (2010), 470–491.
[21] Tom Smith, Phil Husbands, and Michael O'Shea. 2003. Local evolvability of statistically neutral GasNet robot controllers. *Biosystems* 69, 2-3 (2003), 223–243.
[22] Kenneth O Stanley and Risto Miikkulainen. 2002. Evolving neural networks through augmenting topologies. *Evolutionary computation* 10, 2 (2002), 99–127.
[23] Danesh Tarapore, Jeff Clune, Antoine Cully, and Jean-Baptiste Mouret. 2016. How do different encodings influence the performance of the MAP-Elites algorithm?. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, 173–180.
[24] Danesh Tarapore and Jean-Baptiste Mouret. 2015. Evolvability signatures of generative encodings: beyond standard performance benchmarks. *Information Sciences* 313 (2015), 43–61.
[25] Günter P Wagner and Lee Altenberg. 1996. Perspective: complex adaptations and the evolution of evolvability. *Evolution* 50, 3 (1996), 967–976.
[26] Bryan Wilder and Kenneth Stanley. 2015. Reconciling explanations for the evolution of evolvability. *Adaptive Behavior* 23, 3 (2015), 171–179.
[27] Sewall Wright. 1932. The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In *Proceedings of the Sixth International Congress on Genetics*. 356–366.