Procedural Content Generation through Quality Diversity

Daniele Gravina[1], Ahmed Khalifa[2], Antonios Liapis[1], Julian Togelius[2], Georgios N. Yannakakis[1]
1: Institute of Digital Games, University of Malta, Msida, Malta
2: Game Innovation Lab, New York University, NY, USA

***Abstract*—Quality-diversity (QD) algorithms search for a set of good solutions which cover a space as defined by behavior metrics. This simultaneous focus on quality and diversity with explicit metrics sets QD algorithms apart from standard single- and multi-objective evolutionary algorithms, as well as from diversity preservation approaches such as niching. These properties open up new avenues for artificial intelligence in games, in particular for procedural content generation. Creating multiple systematically varying solutions allows new approaches to creative human-AI interaction as well as adaptivity. In the last few years, a handful of applications of QD to procedural content generation and game playing have been proposed; we discuss these and propose challenges for future work.**

***Index Terms*—Procedural Content Generation, Quality Diversity, Evolutionary Computation, Expressivity.**

I. Introduction

Since *ROGUE* (Toy and Wichman, 1980) and *Elite* (Acornsoft, 1984) in the 1980s, certain genres of digital games have relied on algorithmic processes to generate content such as levels, weapons, personalities, quests, etc. Throughout its long history, procedural content generation (PCG) has aimed to provide content that is playable, of a high quality, and yet different from other content that came before or after. On the one hand, most game content need to satisfy certain minimal criteria on playability (such as the exit in a dungeon being reachable by the player) while they also need to be entertaining and challenging (which are softer and often subjective quality dimensions). Content which do not satisfy these criteria of quality can break the gameplay explicitly or implicitly, resulting in a poor player experience. On the other hand, games that rely on PCG to produce fresh content promise that every playthrough, opened chest, or visited settlement will be different. Players expect novel and unseen content at every possible moment, and can swiftly turn against a game where the variation in generated content is low or cosmetic. The backlash against *No Man's Sky* (Hello Games, 2016) was in no small part due to the lack of perceptible variety in the generated worlds [1]. While low-quality generated content can at best affect players' enjoyment and at worst make a game unbeatable, generated content with insufficient diversity can lead to fatigue and dejection. If we look at the large body of research in computational creativity, we can see that value and novelty are vital criteria to evaluate an artifact as creative [2]. While value is commonly targeted in traditional PCG, novelty—or more broadly, game content diversity—is not.

978-1-7281-1884-0/19/$31.00 ©2019 IEEE

In sum, it has been established that many PCG problems require both *quality* and *diversity* of the generated content [3]. This poses a challenge for many existing PCG methods, which are often forced to make a trade-off between these two requirements. In this paper, we argue that recent advancements in search methods allow us to overcome this problem and create PCG algorithms that emphasize quality and diversity simultaneously. This has the potential to significantly increase the scope of what PCG can do in games.

Quality-Diversity (QD) algorithms are a novel family of evolution-like algorithms that simultaneously maintain the quality and diversity of their solutions by rewarding divergence (as novelty or surprise of the artifacts being generated) while maintaining control of the solutions' quality through hard constraints or local competition among individuals with similar behavioral traits. We propose, therefore, procedural content generation through quality-diversity (PCG-QD) as a subset of search-based procedural content generation [4] which is perfectly suited for generating content autonomously (as it can produce a large set of diverse and high-quality artifacts in one run, even in search spaces which are not well-defined) or with a human designer (as it can explain and express its artifacts' desirable properties). This paper presents the components of quality-diversity algorithms, identifies the strengths of PCG-QD over popular alternatives, surveys recent work in this vein and attempts to map out the road ahead.

II. Quality Diversity Approaches

Inspired by the extreme diversity of high-performing creatures found in nature, the *quality diversity* (QD) paradigm [5] has the goal of finding the largest possible set of diverse and high-quality solutions in one run. The search for multiple solutions to a problem is a long-standing challenge in evolutionary computation. In multimodal optimization, for instance, the aim is to find all the local optima of a function by employing niching methods and genotypic diversity mechanisms [6]. However, only searching for local optima does not guarantee the diversity of the solutions, as (behaviorally) diverse solutions may show the same performance measure [7]. Quality diversity, instead, stresses the importance of searching for diverse solutions *first* and then maximize their quality. Indeed, QD draws inspiration from the idea of rewarding divergence to find the necessary steps towards high performing areas of the search space. In divergent search, artificial evolution is not guided by a fitness tied to the ultimate objective of the

problem, but instead rewards directly the diversity of solutions, based on notions such as novelty [8], surprise [9] or curiosity [10]. QD combines the divergent properties of divergent search with localized convergence, as will be discussed below.

A. Divergence Components

As discussed above, QD is based on purely divergent search approaches and thus its core drive is to maintain and reward diversity in a population. There are several ways in which this can be achieved in QD algorithms available today:

1) Behavior Space Distance: Diversity in all known divergence search approaches is measured based on the distance of two individuals in a *behavioral space*. The premise is to have a distance function that can give us a single value that determines how different the behavior of the individual is from the rest of individuals. Unlike other diversity preservation mechanism such as niching [11], divergence is judged on the behavioral space rather than on genotypic similarity. The rest of individuals can be the current population and an archive of past individuals, as in Novelty Search [8], a predicted snapshot of the population as in Surprise Search [9], or explored areas in this individual's lifetime as in Curiosity Search [10].

2) Behavior Space Partitioning: Diversity can also be enforced by partitioning the behavior space using N different dimensions of behavior, where each dimension is discretized and stored as a grid or a *map* of cells. Each cell corresponds to a different area in the behavior space with different properties; this cell contains all individuals that have a certain behaviors. Partitioning could be uniform, as in MAP-Elites [12], or based on the distribution of the population, as in MAP-Elites with Sliding Boundaries [13]. A big difference between the use of a distance function and partitioning the space is that partitioning allows designers to control the granularity of the space.

B. Quality Components

Current QD algorithms have a number of different strategies for pushing evolutionary search towards improving the quality of candidate solutions:

1) Local Competition: A simple way to enhance the quality of the solution is using local competition between individuals within the same niche. To apply this method, a distance metric must be specified to discern which candidates an individual compares itself with. This distance can be the same as the behavior space distance, or could be based on the behavior space partitioning where competing individuals would share the same cell.

2) Constraints: Another way to ensure quality is through a set of constraints that each individual tries to satisfy. These constraints are usually "hard constraints" (e.g. a level can be completed or not), dividing the population into *infeasible* individuals and *feasible* individuals. In most of the previous work [14], [15], [16], [17], infeasible individuals are not considered in terms of their diversity. However, in some cases the diversity is always maintained regardless of constraint satisfaction [18], [19].

C. Algorithms

This subsection discusses the different QD algorithms used in previous work, even if they have not been applied to games.

1) MAP-Elites (ME): MAP-Elites "illuminates" the search space [12], highlighting which attributes of the solutions can contribute to their performance. MAP-Elites combines behavior space partitioning to maintain diversity with local competition to improve the quality by only maintaining the best individual in each cell. The algorithm starts by generating a random set of solutions, which are evaluated in terms of quality and behavioral properties; the latter are used to place this individual in a cell in the partitioned behavior space (feature map). If the chosen cell is empty, the solution is stored in the map; if the cell is occupied, a local competition occurs and the best individual between the two is kept. After this initialization procedure, the stored solutions are uniformly selected to generate a new individual via crossover and/or mutation. The new solution is evaluated and may be stored in the chosen cell if its performance is better than the solution (elite) currently stored there. These phases—selection, mutation, evaluation and replacement—are repeated for a number of evaluations or until the grid reaches a desired coverage.

2) MAP-Elites with Novelty Search (ME-NOV): To bias exploration towards underrepresented solutions in MAP-Elites, [5] used novelty search to select individuals in the feature map. The resulting QD algorithm selects elites to generate offspring proportionally to their novelty score [8]. Offspring are added to the novelty archive, which is used to compute the novelty for each elite stored in the grid. Choosing an appropriate behavioral distance for novelty can bias the areas explored by the algorithm, and a number of variants have been proposed such as combining two feature maps for two behavioral characterizations [5] or searching for novelty in a dimension orthogonal to the features used for space partitioning [20].

3) MAP-Elites with Sliding Boundaries (MESB): Rather than use a predetermined cell size, MESB [13] recomputes the boundaries of the MAP-Elites feature map based on the current distribution of individuals. For every λ individuals generated, the boundaries are recomputed uniformly based on the percentage marks of the distribution along each feature considered. MESB can thus dynamically adapt to unequal distributions, better representing the underlying distribution of the high-performing individuals in the feature space.

4) Constrained Novelty Search (CNS): This QD approach combines the Feasible-Infeasible Two-Population Genetic Algorithm (FI-2Pop) [21] with novelty search, in order to maximize the behavioral space distance of individuals which satisfy certain constraints on quality. CNS maintains two populations, one with only infeasible individuals and one with only feasible. The feasible population evolves to maximize novelty, via novelty search [8], while the infeasible population evolves towards minimizing the distance from feasibility (FINS) or again to maximize novelty (FI2NS) [14]. The generated offspring of two populations can migrate from one population to the other (based on feasibility), which increases the chance to discover stepping stones toward multiple high-quality solutions.

5) Constrained Surprise Search (CSS): Similar to constrained novelty search, this QD algorithm uses unexpectedness as the diversity measure for evaluating feasible individuals in a FI-2pop GA. As in the above variant, CSS is built on the FI-2Pop genetic algorithm, where the infeasible population minimizes the distance from feasibility and the feasible population maximizes surprise through surprise search [9], [22].

6) Constrained MAP-Elites (CME): Combining the constraint-satisfaction capabilities of FI-2Pop with MAP-Elites, constrained MAP-Elites (CME) [18], [19] maintains two populations for each cell of the archive, one containing infeasible individuals and one containing feasible individuals. As in the original formulation of FI-2Pop [21], the feasible population maximizes the predefined fitness, while the infeasible population minimizes the distance from feasibility. Every chromosome is stored in the corresponding cell and population, and the algorithm benefits from the unique features of illumination and constraint satisfaction.

7) Novelty Search with Local Competition (NS-LC): Combining optimization towards a behavior space distance with quality control via local competition, NS-LC [23] applies a multi-objective approach to maximize both the novelty score and a local competition objective which pushes the individual to outperform others in its niche. Novelty search [8] rewards solely the solutions' novelty and ignores the objective of the problem, selecting solutions based on their behavior space distance from other solutions in the population as well as past novel solutions stored in the *novelty archive*. Local competition rewards those individuals that perform better compared to their K nearest neighbours. Local competition ensures a localized convergence considering only the nearest neighbors in the current generation and the novelty archive, creating a local selection pressure across the entire search space.

8) Surprise Search with Local Competition (SS-LC): Similarly to NS-LC, this QD algorithm uses unexpectedness rather than novelty to select individuals in one objective, and local competition as another objective in a multi-objective fashion. Unexpectedness in surprise search [9] is measured as deviation from predictions made on past generations, which offers a different measure of divergence than novelty which rewards unseen solutions [22]. However, surprise search can also be combined with novelty search and local competition, which leads to good performance in highly deceptive domains [22].

III. Why Quality Diversity?

This section highlights what makes PCG-QD an important contribution to the panorama of PCG research. Since all QD approaches surveyed in Section II are based on artificial evolution, PCG-QD is a subset of search-based PCG [4]. However, several key properties of QD algorithms make them better suited than 'typical' SBPCG approaches. Moreover, PCG-QD is compared with popular approaches for generating games in academia and in the industry, namely machine learning (PCGML) [24] and constructive algorithms [25] respectively.

A. Generative Efficiency

The ability of QD approaches to produce a large set of high-quality solutions which exhibit diverse behaviors in one run makes it surprisingly efficient when a broad variety of content is needed. By comparison, constructive algorithms may be computationally fast but output a single artifact; moreover, re-running the algorithm does not ensure that the new output will be particularly different than previous ones. This lack of originality in generated content has been lamented in several games such as *No Man's Sky* (Hello Games, 2015). PCGML similarly outputs a single artifact, and the lack of diversity from one run to the next is even more obvious as the generated content attempt to explicitly follow the same patterns. Traditional SBPCG approaches, while evolving a population of artifacts, are usually interested in the fittest individual at the end of an evolutionary run. Other generative approaches which are fast in producing a feasible individual, such as declarative programming [26], have no way of controlling the diversity of their output. Therefore, while QD can be computationally heavy, a single run can output a vast corpus of high-quality content; this removes the burden of re-running and post-generation assessment of both quality and diversity.

B. Fitness-Free Search

The underlying assumption of search-based approaches [4] is that by defining the fitness function, we can generate content of high quality based on the designer's definition of a *quality measure*. However, defining an effective objective function is not an easy task, exacerbated by known problems such as deceptive fitness landscapes [27], [8]. Furthermore, when the objective of the problem to solve depends on subjective criteria, as is often the case for games, it is difficult to formalize the value to optimize [28]. It is widely recognized that designing a fitness function that incorporates (a) subjective and (b) multiple criteria is a challenging, especially considering that games are multifaceted [29]. Due to their multifaceted nature, an algorithm designer might need to consider both non-functional properties, such as the aesthetic properties and functional properties, e.g., playable levels. While in the literature several solutions have addressed this [4], [3], we argue that quality-diversity fits particularly well as an answer to this problem. Quality-diversity can consider *more than one* dimensions of interest and at the same time explores the fitness landscape based on local rather than global competition in terms of a fitness function. Combined with the explainability of QD approaches (see Section III-E), this can highlight potential biases in the chosen fitness function.

C. Online Expressivity Analysis

A unique feature of quality-diversity as a procedural content generator is the *online expressivity analysis* granted as a byproduct of the search for highly diverse and high-quality solutions. Expressivity analysis [30] is defined as the analysis of the output in terms of styles and variety of artifacts generated by the chosen approach, which can highlight biases of the generator towards specific types of content. While all

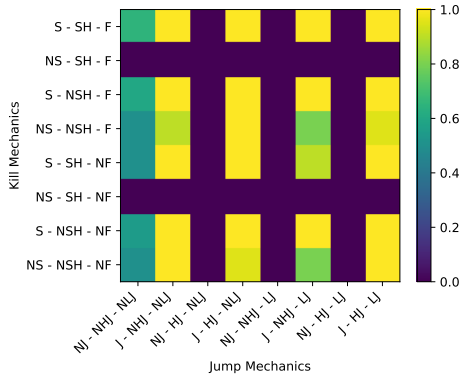

Fig. 1: Expressivity as a heatmap of six binary features (combinations of used mechanics) for Mario scenes in [19].

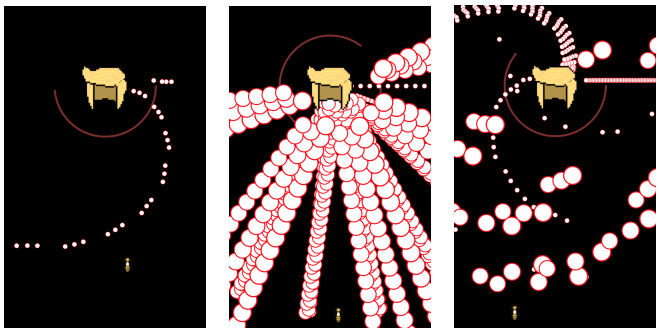

Fig. 2: Bullet-hell level created with constrained MAP-Elites. Figures reproduced with permission from the authors [18].

generators can produce (with multiple re-runs) the large set of artifacts required to perform an expressivity analysis, as noted in Section III-A the QD approaches produce such sets in a single run. More importantly in this context, the diversity components of these QD algorithms perform an expressivity analysis *during* evolution (i.e. online). Moreover, QD algorithms which explicitly optimize for behavioral distance explicitly attempt to increase the expressivity of the generator by searching under-explored niches.

Figure 1 shows an example of the covered space from the work of Khalifa *et al.* [19] on generating scenes in *Super Mario Bros* (Nintendo, 1985), elaborated in Section IV. The figure shows that no levels are generated for some areas of the search space because the divergence characterizations chosen are dependent on each other. For example: a player can not kill an enemy in *Super Mario Bros* without at least jumping. This problem of behavior characterization (BC) dependency will be discussed in more detail in Section V.

D. Human-Machine Co-Creation

Game development often involves design iterations that can completely change the objectives and design priorities. When game developers work alongside an AI-assisted tool [31], [15], the tool's ability to illuminate the space with multiple different and good solutions can help designers identify new designs or to perfect generated content based on their current priorities. QD approaches are able to efficiently (see Section III-A) produce a diverse set of high-quality content, and give a designer control to adjust both the criteria of quality and the behavioral characterization of the artifacts. This makes QD approaches especially effective tools for mixed-initiative content design, and can foster their users' creativity with expected or unexpected but always high-quality suggestions [32].

E. Explainability

Explainable AI for Designers [33] is an important research area that aims to aid the game designer in understanding AI algorithms applied to games. Such explainability is useful during co-creative tasks (see Section III-D) but also for debugging purposes or when the generated artifact is used in another design iteration. Since QD approaches such as MAP-Elites *illuminate* the search space and visualize it as a feature map, this can help developers explore and understand the generator's output [34]. Explainability in this vein is tied to the online and embedded expressivity analysis of these algorithms (see Section III-C). As with any evolutionary algorithm in the SBPCG family, QD approaches can explain the origins of the artifact by showing the lineage of any individual (e.g., in [35], [36]). Strengthening this latter form of explanation, the fact that PCG-QD is efficient in producing a set of good and diverse artifacts in one run (see Section III-A) strengthens this lineage visualization as the common ancestors of different sets of high-performing content can be shown. These different visualizations can help the designer group individuals and select those with the desired features among the many individuals generated by the algorithm.

IV. Cases of Quality Diversity in PCG

This section discusses previous work in PCG-QD, highlighting the differences and commonalities between the different algorithms used. Table I examines each case in terms of five components: the QD algorithm, components of Section II, the quality and diversity characterizations, and the type of artifact produced.

A. Generation of 2D and 3D Objects

MAP-Elites was used to generate 2D images [37] and 3D objects [38]. Similar to DeLeNoX [16], this work combines quality-diversity search and machine learning. A Deep Neural Network (DNN) is first trained on classifying real-world images and then combined with MAP-Elites to generate 2D images or 3D objects. The classification output of the DNN distinguishes between 1000 different classes of images, which is used as diversity characterization of the input. Specifically, MAP-Elites uses the classification output of the DNN as a partition of the search space and tries to optimize every bin based on the confidence of the DNN.

B. Generation of Bullet Hell Scripts

A constrained version of MAP-Elites was used to generate levels for bullet hell games [18]. The generated levels are

Algorithm	Components				Characterization		Artifact
	Divergence		Quality		Divergence	Quality	
	D	P	LC	C			
MAP-Elites	-	✓	✓	-	DNN Output	DNN Confidence	2D and 3D objects [37], [38]
MESB	-	✓	✓	-	Mana Distribution	Health Difference	Hearthstone Decks [13]
CME	-	✓	✓	✓	Playthrough Properties	Validity and Playability	Bullet-Hell Scripts [18]
	-	✓	✓	✓	Triggered Mechanics	Playability and Simplicity	Mario Scenes [19]
	-	✓	✓	✓	Linearity, Simmetry, Similarity, and Patterns	Playability, Room properties, and Design patterns	Dungeons [39]
CNS	✓	-	-	✓	Visual Diversity	Playability	Map Sketches [14], [15]
	✓	-	-	✓	Visual Diversity	Believability	Arcade-Style Spaceships [40]
	✓	-	-	✓	DNN Latent Space	Believability	2D Spaceship Hulls [16]
CSS	✓	-	-	✓	Map Locations	Balance and Playability	FPS Weapons [17]
NS-LC	✓	-	✓	-	Block Presence	Complexity	Minecraft-like Structures [41]

TABLE I: Cases of PCG through Quality Diversity. In the components column, *D* stands for Behavior Space Distance, *P* for Behavior Space Partitioning, *LC* for Local Competition and *C* for Constraints.

Fig. 3: Super Mario Bros scenes created with constrained MAP-Elites. Figures reproduced with permission from the authors [19].

represented as a sequence of events in the Talakat description language [18]. The algorithm tries to evolve valid scripts while making sure that the level is playable using an A* algorithm. The authors use features of the agent and the level (agent entropy, agent risk, and bullet distribution) as dimensions of the feature map. Figure 2 shows three different generated levels from this experiment from different areas in the map.

C. Generation of Mario Scenes

Constrained MAP-Elites was used to generate small sections of levels (scenes) for *Super Mario Bros* [19]. The algorithm tries to find the simplest playable scenes, using an A* agent to check for playability. The algorithm uses the triggered game mechanics during the playthrough as binary dimensions for the map (the mechanic is either fired during the playthrough or not) to make sure the generated scenes target different game mechanics. Eight different mechanics were used, thus creating a feature map of 256 cells. The algorithm was able to generate levels in 100 cells on average (see Fig. 1), ranging from levels where no mechanics were fired to levels where all mechanic were fired. Figure 3 shows three different scenes with different degrees of fired mechanics ranging from no fired mechanics on the left to five mechanics being fired on the right.

D. Generation of Hearthstone Decks

MAP-Elites with Sliding Boundaries [13] was used to generate decks for the online card game *Hearthstone* (Blizzard, 2014). The main goal is to generate high-performing decks

Fig. 4: A screenshot taken from the Sentient Sketchbook tool. Figure reproduced with permission from the authors [15].

that vary in their *mana* distribution curves (mana is the cost required to play a card). In order to do so, the fitness is computed as the difference between the two players' health in 200 games; diversity is based on the distribution of the mana curve, codified as the average and variance of the mana distribution in the population.

E. Generation of Map Sketches

Sentient Sketchbook [15] is a mixed-initiative game design tool where the user can design the level of the game and receive in real-time feedback on playability constraints, balance, etc. PCG-QD has the role of inspiring the user with level suggestions through genetic search. In particular, constrained novelty search [14] is used to produce suggestions using the designer's sketch as an initial seed: a combination of FI-2Pop and novelty search guarantees feasible and highly diverse suggestions. Feasibility tests whether the level is playable (e.g. all resources tiles are reachable by every player's base), while diversity is computed as the number of tiles which are different. Figure 4 shows a screenshot of a design session.

F. Generation of Weapons

A constrained approach to generate a number of diverse and functional weapons for competitive First Person Shooter

Fig. 5: Two example weapons created by constrained surprise search. Figures reproduced with permission from the authors [17].

games is introduced in [17]. These weapons need to be usable and balanced but also exhibit surprising behaviors. In order to accomplish that, a constrained surprise search algorithm is devised to generate weapons that are feasible (balanced and usable) and surprising in their behaviors. In particular, two populations are employed, one where infeasible weapons are optimized towards feasibility and another population where the feasible weapons are rewarded based on their unexpectedness. Surprise is computed based on the death location of the agents used to simulate gameplay with the evolved weapons. A heatmap is computed in every generation based on the coordinates of these death locations and a prediction is made via linear regression of the heatmaps computed in the last two generations. Surprise is then computed based on the difference between the individual's death locations and the predicted heatmap. As an example, Figure 5 shows a pair of generated weapons through this QD approach.

G. Generation of Spaceships

Constrained novelty search has been used extensively for generation of spaceships' visuals, driven by constraints on visual quality (such as all shapes being connected) and by divergence either based on pre-authored visual properties [40] or based on emergent visual patterns. For the latter, DeLeNoX [16] generated increasingly diverse spaceships based on a latent representation and the exploration capabilities of novelty search. Specifically, DeLeNoX alternates two key algorithmic phases, exploration and transformation. During the exploration phase, constrained novelty search generates many diverse but feasible two-dimensional spaceships; during the transformation phase, an autoencoder tries to compress all generated solutions in a low-dimensional latent space. The subsequent exploration phase uses the autoencoder's latent space as behavior characterization, and thus the diversity measure is continuously adapted in every cycle of DeLeNoX.

H. Generation of Minecraft-like Structures

Novelty search with local competition [23] was used to generate a number of diverse block structures for the popular video-game *Minecraft* (Mojang, 2011) in [41]. The block structures are evolved via Artificial Neural Networks, which are in turn evolved via NS-LC. In particular, structures' quality is based on their complexity (net number of placed blocks times the height of the tallest block), while the behavior characterization is a binary vector specifying whether there is a block at each location in the simulation.

I. Generation of Dungeons

An interactive variant of constrained MAP-Elites (CME) is proposed in [39] to blend quality-diversity with the mixed-initiative system Evolutionary Dungeon Designer (EDD). EDD is a design tool capable of generating dungeons for adventure games. In this work, CME is augmented with mixed-initiative capabilities and tested in EDD with different pairs of dimension of interest: symmetry and similarity, number of meso-patterns, number of spatial patterns, and linearity. As in the original CME implementation, the infeasible population minimizes the feasibility constraint (playability) while the feasible population maximizes a weighted sum of inventorial aspects of the room and the spatial distribution of design patterns. The user can choose two dimensions of interest and influence the evolutionary process by selecting the favorite solution displayed in the MAP-Elites grid.

J. Discussion

Observing the cases of PCG-QD in Table I, most cases analyzed use constrained optimization to ensure playable content. This is likely due to games being an *ergodic medium* [42], which cannot be used if the generated content is broken. Since assessing playability often requires extensive tests (e.g. via simulations), constrained optimization approaches are particularly well suited for this. In PCG-QD, the quality component satisfies the feasibility constraints, and is enhanced by the search for diverse content performed within the constrained search space. Future research should enhance PCG-QD with a quality component among only feasible individuals, as in [18].

It is also interesting to inspect the motivation for PCG-QD (see Section III) used in the surveyed work of Table I. *Generative efficiency* and *Human-machine Co-creation* are featured in [15]: the former is beneficial for generating multiple suggestions simultaneously in quasi-real-time[1], while the latter allows the tool to assist the user with both diverse and good suggestions. *Fitness-free search* motivates [17] to generate weapons with unexpected uses, such as the mine-layer. Regarding *online expressivity* analysis, we may argue that it covers (in some cases implicitly) all the described cases of PCG-QD. For instance, in [13] the diversity characterization is used to inspect the expressive features of the cards evolved. More importantly, the underlying distribution of the cards' mana plays an active role during the evolutionary search by affecting the boundaries of the space partitioning. Finally, *explainability* to the designer was implicitly provided in EDD [39]; however, to the best of our knowledge, no project has explicitly targeted explainability in the cases surveyed, which hints at a possible direction for future work.

[1]Sentient Sketchbook [15] also uses standard SBPCG to generate only one high-quality suggestion per evolutionary run, requiring 6 threads for 6 suggestions while CNS produces 6 suggestions in 1 thread.

V. Open problems and Outlook

While several instances of PCG-QD have explored the space of possibilities in games and beyond (surveyed in Section IV), there is extensive work to be done in this direction. We highlight some of the challenges and some of the most promising avenues for exploring QD approaches in games.

A core challenge of PCG-QD is the definition of an effective behavior characterization. The first issue relates to the curse of dimensionality, as a multi-dimensional BC can curtail the advantages described in Section III. Exploring diverse solutions in high-dimensional spaces can hinder the performance of the QD algorithm. Specifically, in high-dimensional spaces it can be hard to find *interesting* diversity across all the possible solutions of the search space [37]. A related challenge is the chosen BC partitioning, as the selected granularity may affect the efficiency of the QD approach. A fine-grained partition can make coverage of the space difficult, while a coarse-grained partition might hide interesting solutions. Several ways to address this have been put forward, such as dimensionality reduction in latent spaces [16] or using centroidal Voronoi tessellation to maintain a constant partition of the space regardless of the number of dimensions [43]. However, future work is required to investigate further how to best handle high-dimensional states. Beyond quality-diversity, an interesting direction for future work would be to fuse different PCG approaches, such as PCGML and PCG-QD. As we highlighted in Section IV, some early examples of a fusion between quality-diversity and machine learning have been tested already [16], [37]. However, an interesting future direction would be to exploit the unique capabilities of PCGML (e.g., autonomous generation, data compression and analysis of the content [24]) with the advantages of quality-diversity (see Section III). For instance, machine-learned models of gameplay [44] could be used to improve the computational efficiency of simulation-based PCG-QD solutions. Numerous and long evaluations of content are usually required to test the quality of the generated content both in SBPCG and PCG-QD. However, we can imagine leveraging the abstraction capabilities of the machine-learned model to create surrogate of the simulations; this approach can reach comparable results in fewer evaluations [45].

Another direction for future work in games is applying QD algorithms to debug, diagnose and understand game playing agents. QD algorithms can generate a diverse set of levels that capture different important features of the level itself and an agent's gameplay [13], [19]. By analyzing the outputs, we can understand which behavior has the most effect on the performance of the playing agent. The same idea can be used to improve the generality of reinforcement learning algorithms [46] by generating diverse sets of levels that can be validated to be playable but the agent can't beat them [47].

Beyond procedural content generation, QD can also be used to evolve a group of playing agents that are either different from each other or compatible with each other. This is especially important for agents playing in collaborative games, as we might need a group of agents that are different from each other and at the same time compatible. For instance, MAP-Elites was used to generate a diverse set of high quality agents [48] that play the game *Hanabi* (Antoine Bauza, 2010). Another use for high-quality diverse agents is for testing generated content with different play styles [49]. This can help debug the game to see if the current content is experienced in a similar manner as intended.

While the cases surveyed in Section IV focused mostly on the generation of levels and visuals, it is important to explore how QD algorithms can work with different facets of game content such as rules [50], music [51], etc. These facets come with their own challenges in defining quality or diversity. For example, in rule generation it is easy to define playable games (games that an automated agent can win in a number of steps) but a formula for good games is more difficult to devise. QD approaches can help explore the space of acceptable artifacts with different characteristics, allowing us to understand the effect these characteristics have on the generated artifacts.

VI. Conclusion

In this paper, we distinguished quality-diversity (QD) as a search strategy for search-based procedural content generation. Based on a range of recent applications of QD to games, QD algorithms can produce a large set of diverse (through controllable and often designer-friendly dimensions) and high-quality content (through constraints on playability and/or local competition). This makes PCG-QD particularly efficient in producing many diverse artifacts in one run, which is useful as explainable designer feedback, in a mixed-initiative tool or for expressivity analysis. Based on the current work in this vein, we identified under-explored areas in terms of algorithms and intended uses. Finally, we laid out a vision for the future of the field and the challenges that it will have to overcome.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 programme under grant agreement No 787476. Ahmed Khalifa acknowledges the financial support from NSF grant (Award number 1717324 - "RI: Small: General Intelligence through Algorithm Invention and Selection.").

References

[1] E. Maiberg, "'No Mans Sky' is like 18 quintillion bowls of oatmeal," https://www.vice.com/en_us/article/nz7d8q/no-mans-sky-review, 2016, accessed 13 May 2019.

[2] G. Ritchie, "Some empirical criteria for attributing creativity to a computer program," *Minds and Machines*, vol. 17, no. 1, pp. 67–99, 2007.

[3] M. Preuss, A. Liapis, and J. Togelius, "Searching for good and diverse game levels," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2014.

[4] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne, "Search-based procedural content generation: A taxonomy and survey," *IEEE Trans. on Computational Intelligence and AI in Games*, vol. 3, no. 3, pp. 172–186, 2011.

[5] J. K. Pugh, L. B. Soros, and K. O. Stanley, "Quality diversity: A new frontier for evolutionary computation," *Frontiers in Robotics and AI*, vol. 3, 2016.

[6] M. Preuss, *Multimodal optimization by means of evolutionary algorithms*. Springer, 2015.

[7] A. Cully and Y. Demiris, "Quality and diversity optimization: A unifying modular framework," *IEEE Trans. on Evolutionary Computation*, 2017.

[8] J. Lehman and K. O. Stanley, "Abandoning objectives: Evolution through the search for novelty alone," *Evolutionary computation*, vol. 19, no. 2, 2011.

[9] D. Gravina, A. Liapis, and G. Yannakakis, "Surprise search: Beyond objectives and novelty," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, 2016, pp. 677–684.

[10] C. Stanton and J. Clune, "Curiosity search: producing generalists by encouraging individuals to continually explore and acquire skills throughout their lifetime," *PloS one*, vol. 11, no. 9, 2016.

[11] M. Preuss, "Improved topological niching for real-valued global optimization," in *Applications of Evolutionary Computation*. Springer, 2012, pp. 386–395.

[12] J.-B. Mouret and J. Clune, "Illuminating search spaces by mapping elites," *arXiv preprint arXiv:1504.04909*, 2015.

[13] M. C. Fontaine, S. Lee, L. B. Soros, F. D. M. Silva, J. Togelius, and A. K. Hoover, "Mapping hearthstone deck spaces with MAP-Elites with sliding boundaries," in *Proceedings of The Genetic and Evolutionary Computation Conference*. ACM, 2019.

[14] A. Liapis, G. N. Yannakakis, and J. Togelius, "Constrained novelty search: A study on game content generation," *Evolutionary computation*, vol. 23, no. 1, pp. 101–129, 2015.

[15] ——, "Sentient sketchbook: Computer-aided game level authoring," in *Proceedings of the 8th Conference on the Foundations of Digital Games*, 2013, pp. 213–220.

[16] A. Liapis, H. P. Martínez, J. Togelius, and G. N. Yannakakis, "Transforming exploratory creativity with delenox,," in *ICCC*, 2013, pp. 56–63.

[17] D. Gravina, A. Liapis, and G. N. Yannakakis, "Constrained surprise search for content generation," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*. IEEE, 2016, pp. 1–8.

[18] A. Khalifa, S. Lee, A. Nealen, and J. Togelius, "Talakat: Bullet hell generation through constrained MAP-Elites," in *Proceedings of The Genetic and Evolutionary Computation Conference*. ACM, 2018, pp. 1047–1054.

[19] A. Khalifa, M. C. Green, G. Barros, and J. Togelius, "Intentional computational level design," in *Proceedings of The Genetic and Evolutionary Computation Conference*. ACM, 2019.

[20] D. Gravina, A. Liapis, and G. Yannakakis, "Blending notions of diversity for MAP-Elites," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2019.

[21] S. O. Kimbrough, G. J. Koehler, M. Lu, and D. H. Wood, "On a feasible–infeasible two-population (fi-2pop) genetic algorithm for constrained optimization: Distance tracing and no free lunch," *European Journal of Operational Research*, vol. 190, no. 2, pp. 310–327, 2008.

[22] D. Gravina, A. Liapis, and G. N. Yannakakis, "Quality diversity through surprise," *IEEE Trans. on Evolutionary Computation*, 2019.

[23] J. Lehman and K. O. Stanley, "Evolving a diversity of virtual creatures through novelty search and local competition," in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '11. ACM, 2011, pp. 211–218.

[24] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmgård, A. K. Hoover, A. Isaksen, A. Nealen, and J. Togelius, "Procedural content generation via machine learning (pcgml)," *IEEE Trans. on Games*, vol. 10, no. 3, pp. 257–270, 2018.

[25] N. Shaker, A. Liapis, J. Togelius, R. Lopes, and R. Bidarra, "Constructive generation methods for dungeons and levels," in *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*, N. Shaker, J. Togelius, and M. J. Nelson, Eds. Springer, 2016.

[26] A. M. Smith and M. Mateas, "Answer set programming for procedural content generation: A design space approach," *IEEE Trans. on Computational Intelligence and AI in Games*, vol. 3, no. 3, 2011.

[27] D. E. Goldberg, "Simple genetic algorithms and the minimal, deceptive problem," in *Genetic algorithms and simulated annealing*, ser. Research Notes in Artificial Intelligence, L. Davis, Ed. Pitman, 1987, pp. 74–88.

[28] M. Csikszentmihalyi and I. S. Csikszentmihalyi, *Optimal experience: Psychological studies of flow in consciousness*. Cambridge university press, 1992.

[29] A. Liapis, G. N. Yannakakis, and J. Togelius, "Computational game creativity," in *Proceedings of the Fifth International Conference on Computational Creativity*, 2014.

[30] G. Smith and J. Whitehead, "Analyzing the expressive range of a level generator," in *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*. ACM, 2010.

[31] G. Smith, J. Whitehead, and M. Mateas, "Tanagra: Reactive planning and constraint solving for mixed-initiative level design," *IEEE Trans. on computational intelligence and AI in games*, vol. 3, no. 3, pp. 201–215, 2011.

[32] A. Liapis, G. N. Yannakakis, C. Alexopoulos, and P. Lopes, "Can computers foster human users' creativity? theory and praxis of mixed-initiative co-creativity," *Digital Culture & Education (DCE)*, vol. 8, no. 2, pp. 136–152, 2016.

[33] J. Zhu, A. Liapis, S. Risi, R. Bidarra, and G. M. Youngblood, "Explainable ai for designers: A human-centered perspective on mixed-initiative co-creation," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2018.

[34] M. Cook, J. Gow, and S. Colton, "Danesh: Helping bridge the gap between procedural generators and their output," in *Proceedings of the FDG workshop on Procedural Content Generation*, 2016.

[35] E. J. Hastings, R. K. Guha, and K. O. Stanley, "Automatic content generation in the galactic arms race video game," *IEEE Trans. on Computational Intelligence and AI in Games*, vol. 1, no. 4, pp. 245–263, 2009.

[36] J. Secretan, N. Beato, D. B. DAmbrosio, A. Rodriguez, A. Campbell, J. T. Folsom-Kovarik, and K. O. Stanley, "Picbreeder: A case study in collaborative evolutionary exploration of design space," *Evolutionary Computation journal*, 2011.

[37] A. M. Nguyen, J. Yosinski, and J. Clune, "Innovation engines: Automated creativity and improved stochastic optimization via deep learning," in *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, 2015, pp. 959–966.

[38] J. Lehman, S. Risi, and J. Clune, "Creative generation of 3d objects with deep learning and innovation engines," in *Proceedings of the 7th International Conference on Computational Creativity*, 2016.

[39] A. Alvarez, S. Dahlskog, J. Font, and J. Togelius, "Empowering quality diversity in dungeon design with interactive constrained MAP-Elites," in *Proceedings of the IEEE Conference on Games*. IEEE, 2019.

[40] A. Liapis, "Exploring the visual styles of arcade game assets," in *Proceedings of Evolutionary and Biologically Inspired Music, Sound, Art and Design (EvoMusArt)*. Springer, 2016.

[41] L. B. Soros, J. K. Pugh, and K. O. Stanley, "Voxelbuild: a minecraft-inspired domain for experiments in evolutionary creativity," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2017, pp. 95–96.

[42] E. J. Aarseth, *Cybertext: Perspectives on ergodic literature*. JHU Press, 1997.

[43] V. Vassiliades, K. Chatzilygeroudis, and J.-B. Mouret, "Using centroidal voronoi tessellations to scale up the multidimensional archive of phenotypic elites algorithm," *IEEE Trans. on Evolutionary Computation*, vol. 22, no. 4, pp. 623–630, 2018.

[44] D. Karavolos, A. Liapis, and G. N. Yannakakis, "Using a surrogate model of gameplay for automated level design," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2018.

[45] A. Gaier, A. Asteroth, and J.-B. Mouret, "Data-efficient design exploration through surrogate-assisted illumination," *Evolutionary computation*, vol. 26, no. 3, pp. 381–410, 2018.

[46] N. Justesen, R. R. Torrado, P. Bontrager, A. Khalifa, J. Togelius, and S. Risi, "Illuminating generalization in deep reinforcement learning through procedural level generation," *arXiv preprint arXiv:1806.10729*, 2018.

[47] D. Anderson, M. Stephenson, J. Togelius, C. Salge, J. Levine, and J. Renz, "Deceptive games," in *Applications of Evolutionary Computation*. Springer, 2018, vol. 10784, LNCS, pp. 376–391.

[48] R. Canaan, J. Togelius, A. Nealen, and S. Menzel, "Diverse agents for ad-hoc cooperation in hanabi," in *Proceedings of the IEEE Conference on Games*, 2019.

[49] A. Liapis, C. Holmgård, G. N. Yannakakis, and J. Togelius, "Procedural personas as critics for dungeon generation," in *Applications of Evolutionary Computation*. Springer, 2015, vol. 9028, LNCS.

[50] A. Khalifa, M. C. Green, D. Perez-Liebana, and J. Togelius, "General video game rule generation," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*. IEEE, 2017.

[51] P. Hutchings and J. McCormack, "Adaptive music composition for games," *IEEE Trans. on Games*, 2019, accepted for publication.