KI - Künstliche Intelligenz (2020) 34:7–17
https://doi.org/10.1007/s13218-020-00647-w

TECHNICAL CONTRIBUTION

# From Chess and Atari to StarCraft and Beyond: How Game AI is Driving the World of AI

**Sebastian Risi**[1] · **Mike Preuss**[2]

Received: 23 January 2020 / Accepted: 31 January 2020 / Published online: 18 February 2020


**Abstract**
This paper reviews the field of Game AI, which not only deals with creating agents that can play a certain game, but also with areas as diverse as creating game content automatically, game analytics, or player modelling. While Game AI was for a long time not very well recognized by the larger scientific community, it has established itself as a research area for developing and testing the most advanced forms of AI algorithms and articles covering advances in mastering video games such as StarCraft 2 and Quake III appear in the most prestigious journals. Because of the growth of the field, a single review cannot cover it completely. Therefore, we put a focus on important recent developments, including that advances in Game AI are starting to be extended to areas outside of games, such as robotics or the synthesis of chemicals. In this article, we review the algorithms and methods that have paved the way for these breakthroughs, report on the other important areas of Game AI research, and also point out exciting directions for the future of Game AI.

## 1 Introduction

For a long time, games research and especially research on Game AI was in a niche, largely unrecognized by the scientific community and the general public. Proponents of Game AI research wrote advertisement articles to justify the research field and substantiate the call for strengthening it (e.g. [45]). The main arguments have been these:

- By tackling game problems as comparably cheap, simplified representatives of real world tasks, we can improve AI algorithms much easier than by modeling reality ourselves.
- Games resemble formalized (hugely simplified) models of reality and by solving problems on these we learn how to solve problems in reality.

Both arguments have at first nothing to do with games themselves but see them as a modeling / benchmarking tools. In our view, they are more valid than ever. However, as in many other digital systems, there has also been and still is a strong intrinsic need for improvement because the performance of Game AI methods was in many cases too weak to be of practical use. This could be both in terms of playing strength, or simply because they failed to produce believable behavior [44]. The latter would be necessary to hold up the suspension of disbelief, or, in other words, the illusion to willingly be immersed in a game world.

But what exactly is Game AI? Opinions on that have certainly changed in the last 10–15 years. For a long time, academic research and game industry were largely unconnected, such that neither researchers tackled AI-related problems game makers had nor the game makers discussed with researchers what these problems actually were. Then, in research some voices emerged, calling for more attention for computer Game AI (partly as opposed to board game AI), including Nareyek [52, 53], Mateas [48], Buro [11], and also Yannakakis [88].

Proponents of a change included Alex Champandard in his computational intelligence and games conference (CIG) 2010 tutorial [94] and Youichiro Miyake in his GameOn

✉ Sebastian Risi
sebr@itu.dk

Mike Preuss
m.preuss@liacs.leidenuniv.nl

[1] IT University of Copenhagen and modl.ai, Copenhagen, Denmark

[2] LIACS, Universiteit Leiden, Leiden, The Netherlands

Asia 2012 keynote.[1] At that time, a large part of Game AI research was devoted to board games as Chess and Go, with the aim to create the best possible AI players, or to game theoretic systems with the aim to better understand these.

Champandard and Miyake both argued that research shall try to tackle problems that are actually relevant also for the games industry. This led to a shift in the focus of Game AI research that was further intensified by a series of Dagstuhl meetings on Game AI that started in 2012.[2] The panoramic view [91] explicitly lists 10 subfields and relates them to each other, most of which were not widely known as Game AI at that time, and even less so in the game industry. Most prominently, areas with a focus on using AI for design and production of games emerged, such as procedural content generation (PCG), computational narrative (nowadays also known as interactive storytelling), and AI-assisted game design. Next to these, we find search and planning, non-player character (NPC) behavior learning, AI in commercial games, general Game AI, believable agents, and games as AI benchmarks. A third important branch that came up at that time (and resembles the 10th subfield) considers modeling players and understanding what happens in a running game (game analysis).

The 2018 book on AI and Games [92] shows the pre-game (design/production) during game (game playing) and after-game (player modeling/game analysis)[3] uses of AI together with the most important algorithms behind it and gives a good overview of the whole field. Due to space restrictions, we cannot go into details on developments in each sub-area of Game AI in this work but rather provide an overview over the ones considered most important, including highlighting some amazing recent achievements that for a long time have not been deemed possible. These are mainly in the game playing field but also draw from generative approaches such as PCG in order to make them more robust.

Most of the popular known big recent successes are connected to big AI-heavy IT companies entering the field such as DeepMind (Google), Facebook AI and OpenAI. Equipped with rich computational and human resources, these new players have especially profited from Deep (Reinforcement) Learning to tackle problems that were previously seen as important milestones for AI, successfully tackling difficult problems of human decision making, such as Go, Dota2, and StarCraft.

It is, however, a fairly open question how we can utilize these successes for solving other problems in Game AI and beyond. As it appears to be possible but utterly difficult to transfer whole algorithmic solutions, e.g., for a complex game as StarCraft, to a completely different domain, we may rather see innovative recombinations of algorithms from the recently enriched portfolio in order to craft solutions for new problems.

In the next sections, we start with enlisting some important terms that will be repeatedly used (Sect. 2) before tackling state / action based learning in Sect. 3. We then report on pixel-based learning in Sect. 4. At this point, PCG comes in as a flexible testbed generator (Sect. 5). However, it is also a viable aim on its own to be able to generate content. Very recently, different sources of game information, such as pixel and state information, are given as input to these game-playing agents, providing better methods for rather complex games (Sect. 6). While many approaches are tuned to one game, others explicitly strive for more generality (Sect. 7). Next to game playing and generating content, we also shortly discuss AI in other roles (Sect. 8). We conclude the article with a short overview of the most important publication venues and test environments in Sect. 9 and some reasoning about the expected future developments in Game AI in Sect. 10.

## 2 Algorithmic Approaches and Game Genres

We provide an overview of the predominant paradigms / algorithm types and game genres, focusing mostly on game playing and more recent literature. These algorithms are used in many other contexts of AI and application areas of course, but some of their most popular successes have been achieved in the Game AI field.

**Reinforcement Learning (RL).** In reinforcement learning an agent learns to perform a task through interactions with its environment and through rewards. This is in contrast to supervised learning, in which the agent is directly told the correct action in different states. One of the main challenges in RL is to find a balance between exploitation (i.e. seeking out states that are known to give a high reward) vs. exploration (i.e. trying out something new that might lead to higher rewards in the long run).

[1] http://igda.sakura.ne.jp/sblo_files/ai-igdajp/academic/YMiyake_GameOnAsia_2012_2_25.pdf.

[2] see http://www.dagstuhl.de/12191, http://www.dagstuhl.de/15051, http://www.dagstuhl.de/17471, http://www.dagstuhl.de/19511.

[3] We are aware that this division is a bit simplistic, of course players can be also modeled online or for supporting the design phase. Please consider this a rough guideline only.

**Deep Learning (DL).** Deep learning is a broad term and comes in a variety of different shapes and sizes. The main distinguishing feature of deep learning is the idea to learn progressively higher-level features through multiple layers of non-linear processing. The most prevalent deep learning methods are based on deep neural networks, which are artificial neural networks with multiple different layers (in new neural network models these can be more than 100 layers). Recent advances in computing power, such as more and more efficient GPUs (which were first developed for fast rendering of 3D games), more data, and various training improvements have allowed deep learning methods to surpass the previous state-of-the-art in many domains such as image recognition, speech recognition or drug discovery. LeCun et al. [38] provide a good review paper on this fast-growing research area.

**Deep Reinforcement Learning.** Deep Reinforcement Learning combines reinforcement learning with deep neural networks to create efficient algorithms that can learn directly from high-dimensional sensory streams. Deep RL has been the workhorse behind many of the recent advances in Game AI, such as beating professional players in StarCraft and Dota2. [3] provides a good overview over deep RL.

**Monte Carlo Tree Search (MCTS).** Monte Carlo Tree Search is a fairly recent [14] randomized tree search algorithm. States of the mapped system are nodes in the tree, and possible actions are edges that lead to new states. In contrast to older methods such as alpha-beta pruning, it does not attempt to look at the full tree but uses controlled exploration and exploitation of already obtained knowledge (successful branches are preferred) and often fully randomized playouts, meaning that a game is played until it ends by applying randomized actions. If that takes too long, state value heuristics can be used alternatively. Loss/win information is propagated upwards up to the tree root such that estimations of the win ratio at every node get available for directing the search. MCTS can thus be applied to much larger trees, but provides no guarantees concerning obtaining optimal solutions. [10] is a popular introductory survey.

**Evolutionary Algorithms (EA).** Also known as bio-inspired optimization algorithms, Evolutionary Algorithms take inspiration from natural evolution for solving black-box optimization problems. They are thus applied when classical optimization methods fail or cannot be employed because no gradient or not even numeric objective value information (but ranking of solutions) is available. A key idea of EAs is parallel search by means of populations of candidate solutions, which are concurrently improved, making it a global optimization method. EAs are especially well suited for multi-objective optimization, and the well-known GA, NSGA-II, CMA-ES algorithms are all EAs, see also the introduction/survey book [18].

Which are the most important games to serve as testbeds in Game AI? The research-oriented frameworks general game playing (GGP), general video Game AI (GVGAI) and the Atari learning environment (ALE) play an important role but are somewhat far from modern video games. This also holds true for the traditional AI challenge board games Chess and Go and card games as Poker or Hanabi. In video games, the predominant genres are real-time strategy (RTS) games such as StarCraft, Multiplayer online battle arena (MOBA) games such as Dota2, and first person shooter (FPS) games such as Doom. Sports games currently get more important [43] as they often represent a competitive team situation that is seen as similar to many real-world human/AI collaborative problems. In a similar way, cooperative (capture-the-flag) variants of FPS games [31] are used. Figures 1 and 2 provide an overview of the different properties of the games used as AI testbeds.

## 3 Learning to Play from States and Actions

Games have for a long time served as invaluable testbeds for research in artificial intelligence (AI). In the past, particularly board games such as Checkers and Chess have been tackled, later on turning to Go when Checkers had been solved [70] and with DeepBlue [12] an artificial intelligence had defeated the world champion in Chess consistently. All these games and many more, up to Go, have one thing in common: they can be expressed well by states and actions, where the number of actions is usually a not-too-large number of often around 100 or less reasonable moves from any possible position. For quite some time, board games have been tackled with alpha-beta pruning (Turing Award Winners Newell and Simon explain in [54] how this idea came up several times almost at once) and very sophisticated and extremely specialized heuristics before Coulom invented

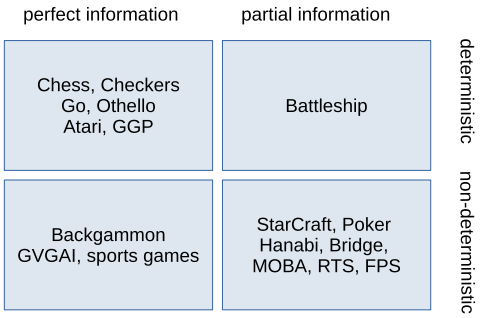


**Fig. 1** Available information and determinism as separating properties for different games treated in Game AI

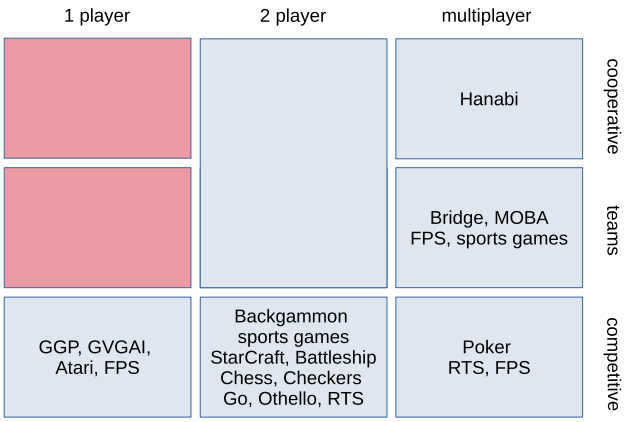


**Fig. 2** Player numbers and style from cooperative to competitive for different games or groups of games treated in Game AI. Note that for several games, multiple variants are possible, but we use only the most predominant ones

Monte Carlo Tree Search (MCTS) [14] in 2006. MCTS gives up optimality (full exploration) in exchange for speed and is therefore now dominating AI solutions for larger board games such as Go with about $10^{170}$ possible states (board positions). MCTS-based Go algorithms had greatly improved the state-of-the-art up to the level of professional players by incorporating sophisticated heuristics as Rapid Action Value Estimation (RAVE) [21]. In the following, MCTS based approaches were shown to cope well also with real-time conditions as in the PacMan game [59] and also hidden information games [62].

However, only the combination of MCTS with DL led to a world-class professional human-level Go AI player named AlphaGo [76]. At this stage, human experience (recorded grandmaster games) had been used for "seeding" the learning process that was then accelerated by self-play. By playing against itself, the AlphaGo algorithm was able to steadily improve its value (how good is the current state?) and policy (what is the best action to play?) artificial neural networks. The next step, AlphaGo Zero [77] removed all human data, relying on self-play alone, and learned to play Go better than the original AlphaGo approach but from scratch. This approach has been further developed to AlphaZero [75] and shown to be able to learn to play different games, next to Go also Chess and Shogi (Japanese Chess). In-depth coverage of most of these developments is also provided in [61].[4]

From the last paragraphs, it may appear as if learning via self-play is limited to two-player perfect information games only. However, also multi-player partial information games such as Poker [9] and even cooperative multi-player games such as Hanabi [39] have recently been tackled and AI players now exist that can play these games at the level of the best human players. Thus, is self-play the ultimate AI solution for all games? Seemingly not, as [85] suggests (see Sect. 6). However, this may be a question of the number of actions and states in a game and remains to be seen. Nevertheless, board games and card games are obviously good candidates for such AI approaches.

## 4 Learning to Play from Pixels

For a long time, learning directly from high-dimensional input data such as the pixels of a video game was an unsolved challenge. Earlier neural network-based approaches for playing games such as Pac-Man relied on careful engineered features such as the distance to the nearest ghost or pill, which are given as input to the neural network [67].

While some earlier game-playing approaches, especially from the evolutionary computation community, showed initial success in learning directly from pixels [20, 29, 57, 82], it was not until DeepMind's seminal paper on learning to play Atari video games from pixels [50, 51] that these approaches started to compete and at times outperform human players. Serving as a common benchmark, many novel AI algorithms have been developed and compared on Atari video games first [33] before being applied to other domains such as robotics [1]. A computationally cheap and thus interesting end-to-end pixel-based learning environment is VizDoom [36], a competition setting that relies on a rather old game that is run in very small screen resolutions. Low resolution pixel inputs are also employed in the obstacle tower challenge (OTC) [32].

DeepMind's paper ushered in the area of *Deep Reinforcement Learning*, combining reinforcement learning with a rich neural network-based representation (see infobox for more details). Deep RL has since established itself as the prevailing paradigm is to learn directly from high-dimensional input such as images, videos, or sounds

[4] https://learningtoplay.net/.

without the need for human-design features or preprocessing. More recently, approaches based on evolutionary algorithms have shown to also be competitive with approaches based on gradient descent-based methods [80].

However, some of the Atari games, namely Montezuma's Revenge, Pitfall, and others proved to be too difficult to solve with standard deep RL approaches [50] because of sparse and/or late rewards. These hard-exploration games can be handled successfully by evolutionary algorithms that explicitly favor exploration such as Go-Explore [17].

A recent trend in deep RL is to allow agents to learn a general model of how their environment behaves and use that model to explicitly plan ahead. For games, one of the first approaches was the World Model introduced by [26], in which an agent learns to solve a challenging 2D car racing game and a 3D VizDoom environment from pixels alone. In this approach, the agent first learns by collecting observations from the environment, and then training a forward model that takes the current state of the environment and action and tries to predict the next state. Interestingly, this approach also allowed an agent to get better by training inside a hallucinated environment created through a trained world model.

Instead of first training a policy on random rollouts, follow-up work showed that end-to-end learning through reinforcement learning [28] and evolution [65, 66] is also possible. We will discuss MuZero as another example of planning in latent space in Sect. 6.

## 5 Procedural Content Generation

In addition to playing games, another active area of AI research is procedural content generation (PCG) [68, 74]. PCG refers to the algorithmic creation of game content such as levels, textures, quests, characters, or even the rules of the game itself.

One of the appeals of employing PCG in games is that it can increase their replayability by offering the player a new experience every time they play. For example, games such as No Man's Sky (Hello Games, 2016) or Spelunky (Mossmouth, LLC, 2013) famously featured PCG as part of their core gameplay, allowing players to explore an almost unlimited variety of planets or caves. One of the most important early benefits of PCG methods was that it allowed the creation of larger game worlds than what would normally fit on a computer's hard disk at the time. One of the first games using PCG-based methods was *Elite* (Brabensoft, 1984), a space trading video game featuring thousands of planets. The whole starsystem with each visited planet and space stations could be recreated from a given random seed.

While the origin of PCG is rooted in creating a more engaging experience for players [93], more recently PCG-based approaches have also found important other use cases. With the realisation that methods such as deep reinforcement learning can surpass humans in many games, also came the realisation that these methods overfit to the exact environment they are trained on [35, 96]. For example, an agent trained to reach the level of a human expert in a game such as Breakout, will fail completely when tested on a Breakout version where the game pedal has a slightly different size or is at a slightly different position. Recent research showed that by training agents on many procedurally generated levels allows them to become significantly more general [35]. In an impressive extension of this idea, DeepMind trained agents on a large number of randomly created levels to reach human-level performance in the Quake III Capture the Flag game [31]. This trend to make AI approaches more general by training them on endless variations of environments was continued in the hide-and-seek work by OpenAI [4] and also in the obstacle tower challenge (OTC) [32] and will certainly also be employed in many future approaches.

Meanwhile, PCG has been applied to many different types of game components or facets (e.g. visuals, sound), but most often to only one of these at once. One of the open research questions in this context is how generators for different facets can be combined [41].

Similar to some of the other techniques described in this article, PCG has also more recently found to be applicable to areas outside of games [68]. For example, training a humanoid robot hand to manipulate a Rubik's cube in a simulator on many variants of the same problem (e.g. varying parameters such as the size, mass, and texture of the cube) has allowed a policy trained in a simulator to sometimes work on a physical robot hand in the real world. For a review of how PCG has increased generality in machine learning we refer the interested reader to this survery [68] and for a more in-depth review of PCG in general to the book by Shaker et al. [74].

## 6 Merging State and Pixel Information

Whereas the AI in AlphaGo and its predecessors for playing board games dealt with board positions and possible moves, deep RL and recent evolutionary approaches for optimising deep neural networks (a research field now referred to as deep neuroevolution [79]), learn to play Atari games directly from pixel information. On the one hand, these approaches have some conceptual simplicity, but on the other hand, it is intuitively clear that adding more information—if available—may be of advantage. More recently, these two ways of obtaining game information were joined in different ways.

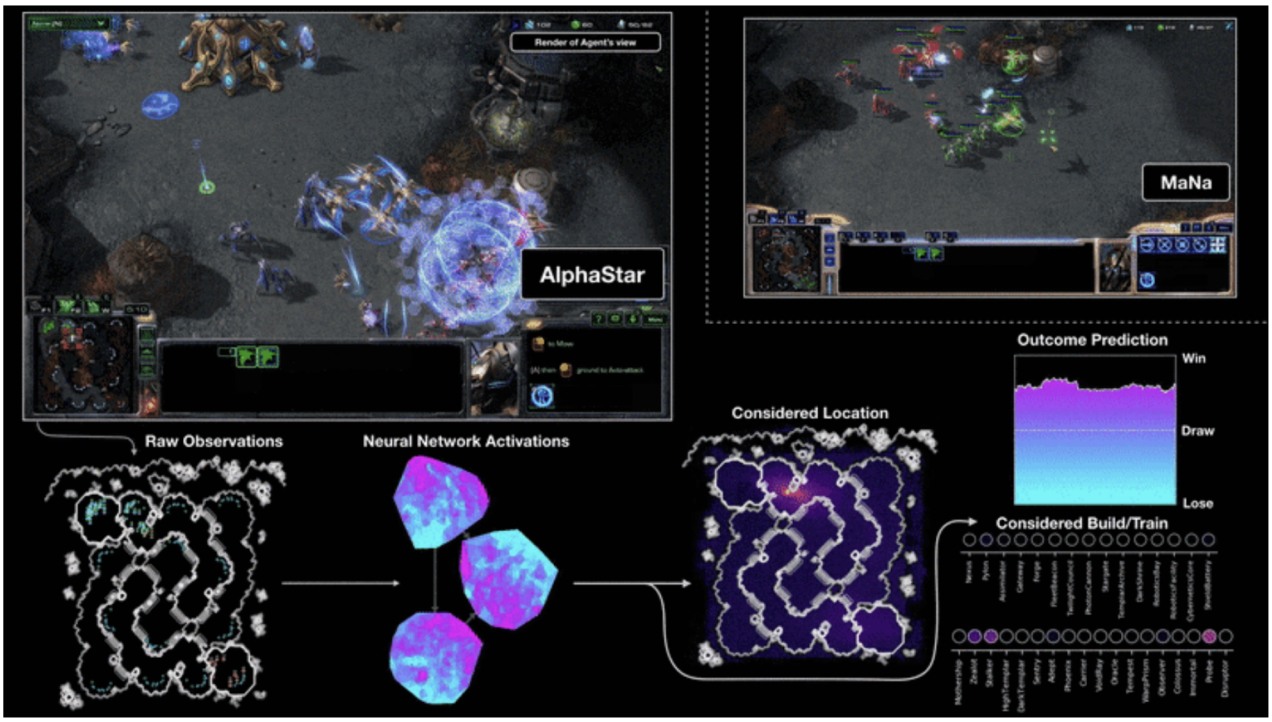


**Fig. 3** A visualisation of the AlphaStar agent playing against the human player MaNa, from [84]. Shown is the raw observation that the neural network gets as input (bottom left), together with the internal neural network activations. On the lower right side are shown actions considered by the agent together with a prediction of the outcome of the game

The hide-and-seek approach [4] depends on visual and state information of the agents but also heavily on the use of co-evolutionary effects in a multi-agent environment that very much reminds of EA techniques.

In AlphaStar (Fig. 3) that was designed to play StarCraft at human professional level, both state information (location and status of units and buildings) as well as pixel information (minimap) is fed into the algorithm. Interestingly, self-play is used heavily, but is not sufficient to generate human professional competitive players because the strategy space is huge and human opponents may come up with very different ways to play the game that must all be handled. Therefore, as in AlphaGo, human game data is used to seed the algorithm. Furthermore, also co-evolutionary effects in a 3 tier league of different types of agents are driving the learning process. It shall be noted that the success of AlphaStar was hard to imagine only some years ago because RTS games were considered the hardest possible testbeds for AI algorithms in games [55]. These successes are, however, not without controversy and people argue if the comparisons of AIs playing against humans are fair [13, 34].

MuZero [71] is able to learn playing Atari games (pixel input) as well as Chess and Go (state input) by generating virtual states according to reward/position value similarity. These are managed in a tree-like fashion as in MCTS but costly rollouts are avoided. The elegance of this approach lies in the ability to use different types of input and the construction of an internal representation that is oriented only at values and not at exact game states.

## 7 Towards More General AI

While AI algorithms have become exceedingly good at playing specific games [33], it is still an unsolved challenge how to make an AI algorithm that can learn to quickly play any game it is given, or how to transfer skills learned in one game to another. This challenge, also known as General Video Game Playing [22], has resulted in the development of the General Video Game AI framework (GVGAI), a flexible framework designed to facilitate the development of general AI through video game playing [60].

With increasingly complicated worlds and graphics, video games might be the ideal environment to learn more general intelligence. Another benefit of games is that they often share similar controllers and goals. To spur developments in this area, the GVGAI framework now also includes a Learning Track, in which the goal of the agent is to learn a new game quickly without being trained on it beforehand. The hope is that methods that can quickly learn any game

they are given, will also ultimately be able to quickly learn other tasks such a robot manipulation in the real world.

Whereas most successful approaches for GVGAI games employ MCTS, it shall be noted that there are also other competitive approaches as the rolling horizon evolutionary algorithm (RHEA) [42] that evolve partial action sequences as a whole through an evolutionary optimization process. Furthermore, DL variants start to get used here as well [83].

## 8 AI for Player Modelling and Other Roles

In this section, we briefly mention a few other use cases for current AI methods. In addition to learning to play or generating games and game content, another important aspect of Game AI—and potentially currently the main use case in the game industry—is game analytics. Game analytics has changed the game landscape dramatically over the last ten years. The main idea in game analytics is to collect data about the players while they play the game and then update the game on the fly. For example, the difficulty of levels can be adjusted or the user interface can be streamlined. At what point players stopped playing the game can be an important indication of what to change to reduce the game's churn[5] rate [27, 37, 69]. We refer the interested reader to the book on game analytics by El-Nasr et al. [19].

Another important application area of Game AI is player modelling. As the name suggests, player modelling aims to model the experience or behavior of the player [5, 95]. One of the main motivations for learning to model players is that a good player model can allow the game to be tailored even more to the individual player. A variety of different approaches to model players exist, such as supervised learning (e.g. training a neural network in a supervised way on recorded plays of human players to behave the same way), to unsupervised approaches such as clustering that aim to group similar players together [16]. Based on which cluster a new player belongs to, different content or other game adaptations can be performed. Combining PCG (Sect. 5) with player modelling, an approach called Experience-Driven Procedural Content Generation [93], allows these algorithms to automatically generate unique content that induces a desired experience for a player. For example, [58] trained a model on players of Super Mario, which could then be used to automatically generate new Mario levels that maximise the modelled fun value for a particular player. Exciting recent work can even predict a player's affect in certain situation from pixels alone [47].

There is also a large body of research on human-like non-player characters (NPC) [30], and some years ago, this research area was at the core of the field, but with the upcoming interest in human/AI collaboration it is likely to thrive again in the next years.

Other roles for Game AI include playtesting and balancing which both belong to game production and mostly happen before games are published. Testing for bugs or exploits in a game is an interesting application area of huge economic potential and some encouraging results exist [15]. With the rise of machine learning methods that can play games at a human or beyond human level and methods that can solve hard-exploration games such as Montezuma's Revenge [17], this area should see a large increase of interest from the game industry in the coming years. Mixed-initiative tools that allow humans to create game content together with a computational creator often include an element of automated balancing, such as balancing the resources on a map in a strategy game [40]. Game balancing is a wide and currently under-researched area that may be understood as a multi-instance parameter tuning problem. One of the difficulties here is that many computer games do not allow headless accelerated games and APIs for controling these. Some automated approaches exist for single games [63] but they usually cannot cope with the full game and approaches for more generally solving this problem are not well established yet [86]. Dynamic re-balancing during game runtime is usually called dynamic difficulty adaptation (DDA) [78].

## 9 Journals, Conferences, and Competitions

The research area of Game AI is centered in computer science, but influenced by other disciplines as i.e. psychology, especially when it comes to handling humans and their emotions [89, 90]. Furthermore, (computational) art and creativity (for PCG), game studies (formal models of play) and game design are important neighboring disciplines.

In computer science, Game AI is not only limited to machine learning and traditional branches of AI but also has links to information systems, optimization, computer vision, robotics, simulation, etc. Some of the core conferences for Game AI are:

- Foundations of Digital Games (FDG)
- IEEE Conference on Games (CoG), until 2018 the Conference on Computational Intelligence and Games (CIG)
- Artificial Intelligence for Interactive Digital Entertainment (AIIDE)

[5] In the game context, churn means that a player who has played a game for some time completely stops playing it. This is usually very hard to predict but essential to know especially for online game companies.

Also, many computer science conferences have tracks or co-located smaller conferences on Game AI, as e.g. GECCO and IJCAI. The more important journals in the field are the IEEE Transactions on Games ToG (formerly TCIAIG) and the IEEE Transactions on Affective Computing. The most active institutes in the area can be taken from a list (incomplete, focused only on the most relevant venues) compiled by Mark Nelson.[6]

A large part of the progress of the last years is due to the free availability of competition environments as: StarCraft, GVGAI, Angry Birds, Hearthstone, Hanabi, MicroRTS, Fighting Game, Geometry Friends and more, and also the more general frameworks as: ALE, GGP, OpenSpiel, OpenAIGym, SC2LE, MuJoCo, DeepRTS.

## 10 The Future of Game AI

More advanced AI techniques are slowly finding their way into the game industry and this will likely increase significantly over the coming years. Additionally, companies are more and more collaborating with research institutions, to bring the latest innovations out to the industry. For example, Massive Entertainment and the University of Malta collaborated to predict the motivations of players in the popular game Tom Clancys The Division [49]. Other companies, such as King, are investing heavily in deep learning methods to automatically learn models of players that can then be used for playtesting new levels quickly [25].

Procedural content generation is already employed for many mainstream games such as Spelunky (Mossmouth, LLC, 2013) and No Man's Sky (Hello Games, 2016) and we will likely see completely new types of games in the future that would be impossible to realise without sophisticated AI techniques. The recent AI Dungeon 2 game (http://www.aidungeon.io) points to what type of direction these games might take. In this text adventure game players can interact with Open AI's GPT-2 language model, which was trained on 40 gigabytes from text scraped from the internet. The game responds to almost anything the player types in a sensible way, although the generated stories also often lose coherence after a while. This observation points to an important challenge: For more advanced AI techniques to be more broadly employable in the game industry, approaches are needed that are more controllable and potentially interpretable by designers [97].

We predict that in the near future, generative modelling techniques from machine learning, such as Generative and Adversarial Networks (GANs) [24], will allow users to personalise their avatars to an unprecedented level or allow the creation of an unlimited variety of realistic textures and assets in games. This idea of *Procedural Content Generation via Machine Learning* (PCGML) [81], is a new emerging research area that has already led to promising results in generating levels for games such as Doom [23] or Super Mario [87].

From the current perspective, we would expect that future research (next to playing better on more games) in Game AI will focus on these areas:

- AI/human collaboration and AI/AI agent collaboration is getting more important, this may be subsumed under the term *team AI*. Recent attempts in this direction include e.g.: Open AI five [64], Hanabi [6], capture the flag [31]
- More natural language processing enables better interfaces and at some point free-form direct communication with game characters. Already existing commercial voice-driven assistance systems as the Google Assistant or Alexa show that this is possible.
- The previous points and the progress in player modeling and game analysis will lead to more human-like behaving AI, this will in turn enable better playtesting that can be partly automated.
- PCG will be applied more in the game industry and other applications. For example, it is used heavily in Microsoft's new flight simulator version that is now (January 2020) in alpha test mode. This will also trigger more research in this area.

Nevertheless, as in other areas of artificial intelligence, Game AI will have to cope with some issues that mostly stem from two newer developments: theory-light but very successful deep learning methods, and highly parallel computation. The first entails that we have very little control over the performance of deep learning methods, it is hard to predict what works well with which parameters, and the second one means that many experiments can hardly ever be replicated due to hardware limitations. E.g., Open AI Five has been trained on 256 GPUs and 128,000 CPUs [56] for a long time. More generally, large parts of the deep learning driven AI are currently presumed to run into a reproducibility crisis.[7] Some of that can be cured by better experimental methodology and statistics as also worked well in Evolutionary Computation some time ago [7]. First attempts in Game AI also try to approach this problem by defining guidelines for experimentation, e.g. for the ALE [46], but replicating experiments that take weeks is an issue that will probably not easily be solved.

It is definitively desired to apply the algorithms that successfully deal with complex games also to other application

[6] http://www.kmjn.org/game-rankings.

[7] https://www.wired.com/story/artificial-intelligence-confronts-reproducibility-crisis/.

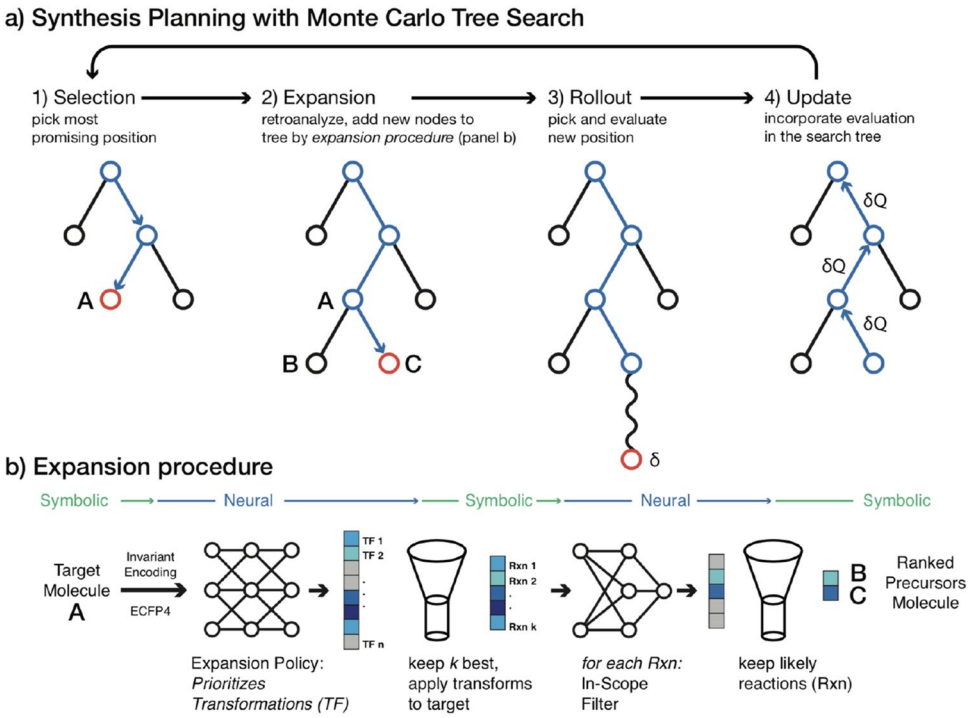


**Fig. 4** Chemical retrosynthesis on basis of the AlphaGo approach; figure from [73]. The upper subfigure shows the usual MCTS steps, and the lower subfigure links these steps to the chemical problem. Actions are now chemical reactions, states are the derived chemical compounds. Instead of preferred moves in a game, the employed neural networks learn reaction preferences. In contrast to AlphaGo, possible moves are not simply provided but have to be learned from data, an approach termed "world program" [72]

areas. Unfortunately, this is usually not trivial, but some promising examples already exist. The AlphaGo approach that is based on searching by means of MCTS in a neural network representation of the treated problem has been transfered to the chemical retrosynthesis problem [73] that consists of finding a synthesis path for a specific chemical component as depicted in Fig. 4. As for the synthesis problem, in contrast to playing Go, the set of feasible moves (possible reactions) is not given but has to be learned from data, the approach bears some similarity to MuZero [71]. The idea to learn a forward model from data has been termed *world program* [72].

Similarly, the same distributed RL system that OpenAI used to train a team of five agents for Dota 2 [8], was used to train a robot hand to perform dexterous in-hand manipulation [2].

We believe Game AI research will continue to drive innovations in the world of AI and hope this review article will serve as a useful guide for researchers entering this exciting research field.

**Acknowledgements** We would like to thank Mads Lassen, Rasmus Berg Palm, Niels Justesen, Georgios Yannakakis, Marwin Segler, and Christian Igel for comments on earlier drafts of this manuscript.

## References

1. Akkaya I et al (2019) Solving Rubik's cube with a robot hand. arXiv:1910.07113
2. Andrychowicz OM et al (2020) Learning dexterous inhand manipulation. Int J Robot Res 39(1):3–20
3. Arulkumaran K et al (2017) A brief survey of deep reinforcement learning. arXiv:1708.05866
4. Baker B et al (2019) Emergent tool use from multi-agent Autocurricula. arXiv:1909.07528
5. Bakkes SC, Spronck PH, van Lankveld G (2012) Player behavioural modelling for video games. Entertain Comput 3(3):71–79
6. Bard N et al (2019) The Hanabi challenge: a new frontier for AI research. In: CoRR abs/1902.00506. arXiv:1902.00506
7. Bartz-Beielstein T et al (2010) Experimental methods for the analysis of optimization algorithms. Springer, Berlin
8. Berner C et al (2019) Dota 2 with large scale deep reinforcement learning. arXiv:1912.06680
9. Brown N, Sandholm T (2019) Superhuman AI for multiplayer poker. Science 365(6456):885–890
10. Browne CB et al (2012) A survey of Monte Carlo tree search methods. IEEE Trans Comput Intell AI Games 4(1):1–43
11. Buro M (2003) Real-time strategy games: a new AI research challenge. In: Gottlob G, Walsh T (eds) IJCAI-03, proceedings of the eighteenth international joint conference on artificial intelligence, Acapulco, Mexico, August 9–15, 2003. Morgan Kaufmann, pp 1534–1535
12. Campbell M, Hoane AJ Jr, Hsu F-H (2002) Deep blue. Artif Intell 134(1–2):57–83
13. Canaan R et al (2019) Leveling the playing field-fairness in AI versus human game benchmarks. arXiv:1903.07008
14. Coulom R (2006) Efficient selectivity and backup operators in Monte-Carlo tree search. In: van den Herik HJ, Ciancarini P, Donkers HHLM (eds) Computers and games, 5th international conference, CG 2006, Turin, Italy, May 29–31, 2006. Revised Papers, vol 4630. Lecture Notes in Computer Science. Springer, pp 72–83
15. Denzinger J et al (2005) Dealing with parameterized actions in behavior testing of commercial computer games. In: CIG. Citeseer
16. Drachen A, Canossa A, Yannakakis GN (2009) Player modeling using self-organization in Tomb Raider: underworld. In: 2009

IEEE symposium on computational intelligence and games. IEEE. pp 1–8
17. Ecoffet et al (2019) A go-explore: a new approach for hard-exploration problems. arXiv:1901.10995
18. Eiben AE, Smith JE (2015) Introduction to evolutionary computing, 2nd edn. Springer, Berlin
19. El-Nasr MS, Drachen A, Canossa A (2016) Game analytics. Springer, Berlin
20. Gallagher M, Ledwich M (2007) Evolving pac-man players: can we learn from raw input? In: 2007 IEEE symposium on computational intelligence and games. IEEE, pp 282–287
21. Gelly S, Silver D (2011) Monte-Carlo tree search and rapid action value estimation in computer Go. Artif Intell 175(11):1856–1875
22. Genesereth M, Love N, Pell B (2005) General game playing: overview of the AAAI competition. AI Mag 26(2):62–62
23. Giacomello E, Lanzi PL, Loiacono D (2018) DOOM level generation using generative adversarial networks. In: IEEE games, entertainment, media conference (GEM). IEEE, pp 316–323
24. Goodfellow I et al (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680
25. Gudmundsson SF et al (2018) Human-like playtesting with deep learning. In: 2018 IEEE conference on computational intelligence and games (CIG). IEEE, pp 1–8
26. Ha D, Schmidhuber J (2018) World models. arXiv:1803.10122
27. Hadiji F et al (2014) Predicting player churn in the wild. In: 2014 IEEE conference on computational intelligence and games. IEEE, pp 1–8
28. Hafner D et al (2018) Learning latent dynamics for planning from pixels. arXiv:1811.04551
29. Hausknecht M et al (2014) A neuroevolution approach to general atari game playing. IEEE Trans Comput Intell AI Games 6(4):355–366
30. Hingston P (2012) Believable bots: can computers play like people?. Springer, Berlin
31. Jaderberg M et al (2019) Human-level performance in 3D multiplayer games with population-based reinforcement learning. Science 364(6443):859–865
32. Juliani A et al (2019) Obstacle tower: a generalization challenge in vision, control, and planning. In: Kraus S (ed)Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI 2019, Macao, China, August 10–16, 2019. http://ijcai.org, pp 2684–2691
33. Justesen N et al (2019) Deep learning for video game playing. IEEE Trans Games 1–1
34. Justesen N, Debus MS, Risi S (2019) When are we done with games? In: 2019 IEEE conference on games (CoG). IEEE, pp 1–8
35. Justesen N et al (2018) Illuminating generalization in deep reinforcement learning through procedural level generation. In: arXiv :1806.10729
36. Kempka M et al (2016) Vizdoom: a doom-based ai research platform for visual reinforcement learning. In: 2016 IEEE conference on computational intelligence and games (CIG). IEEE, pp 1–8
37. Kummer LBM, Nievola JC, Paraiso EC (2018) Applying commitment to churn and remaining players lifetime prediction. In: 2018 IEEE conference on computational intelligence and games, CIG, Maastricht, The Netherlands, August 14–7, 2018. IEEE, 2018, pp 1–8
38. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436
39. Lerer A et al (2019) Improving policies via search in cooperative partially observable games. arXiv:1912.02318 [cs.AI]
40. Liapis A, Yannakakis GN, Togelius J (2013) Sentient Sketchbook: computer-aided game level authoring. In: FDG, pp 213–220
41. Liapis A et al (2019) Orchestrating game generation. IEEE Trans Games 11(1):48–68
42. Liebana DP et al (2013) Rolling horizon evolution versus tree search for navigation in single-player real-time games. In: Blum C, Alba E (eds) Genetic and evolutionary computation conference, GECCO '13, Amsterdam, The Netherlands, July 6–10, 2013. ACM, pp 351–358
43. Liu S et al (2019) Emergent coordination through competition. In: International conference on learning representations
44. Livingstone D (2006) Turing's test and believable AI in games. Comput Entertain 4:1
45. Lucas SM, Kendall G (2006) Evolutionary computation and games. IEEE Comput Intell Mag 1(1):10–18
46. Machado MC et al (2017) Revisiting the arcade learning environment: evaluation protocols and open problems for general agents. In: CoRR abs/1709.06009 . arXiv:1709.06009
47. Makantasis K, Liapis A, Yannakakis GN (2019) From pixels to affect: a study on games and player experience. In: 2019 8th international conference on affective computing and intelligent interaction (ACII). IEEE, pp 1–7
48. Mateas M (2003) Expressive AI: games and artificial intelligence. In: DiGRA & # 3903—Proceedings of the 2003 DiGRA International Conference: Level Up
49. Melhart D et al (2019) Your gameplay says it all: modelling motivation in Tom Clancy's the division. arXiv:1902.00040
50. Mnih V et al (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529
51. Mnih V et al (2013) Playing Atari with deep reinforcement learning. In: CoRR arxiv:abs/1312.5602
52. Nareyek A (2007) Game ai is dead. Long live game ai!. IEEE Intell Syst 22(1):9–11
53. Nareyek A (2001) Review: intelligent agents for computer games. In: Marsland T, Frank I (eds) Computers and games. Springer, Berlin Heidelberg, pp 414–422
54. Newell A, Simon HA (1976) Computer science as empirical inquiry: symbols and search. Commun ACM 19(3):113–126
55. Ontãnón S et al (2013) A survey of real-time strategy game AI research and competition in StarCraft. IEEE Trans Comput Intell AI Games 5(4):293–311
56. OpenAI. OpenAI Five. https://blog.openai.com/openai-five/. (2018)
57. Parker M, Bryant BD (2012) Neurovisual control in the Quake II environment. IEEE Trans Comput Intell AI Games 4(1):44–54
58. Pedersen C, Togelius J, Yannakakis GN (2010) Modeling player experience for content creation. IEEE Trans Comput Intell AI Games 2(1):54–67
59. Pepels T, Winands MHM, Lanctot M (2014) Real-Time Monte Carlo Tree Search in Ms Pac-Man. IEEE Trans Comput Intell AI Games 6(3):245–257
60. Perez-Liebana D et al. (2016) General video game ai: competition, challenges and opportunities. In: Thirtieth AAAI conference on artificial intelligence
61. Plaat A (2020) Learning to play—reinforcement learning and games. https://learningtoplay.net/
62. Powley EJ, Cowling PI, Whitehouse D (2014) Information capture and reuse strategies in Monte Carlo Tree Search, with applications to games of hidden information. Artif Intell 217:92–116
63. Preuss et al. M (2018) Integrated balancing of an RTS game: case study and toolbox refinement. In: 2018 IEEE conference on computational intelligence and games, CIG 2018, Maastricht, The Netherlands, August 14–17, 2018. IEEE, pp 1–8
64. Raiman J, Zhang S, Wolski F (2019) Long-term planning and situational awareness in OpenAI five. In: arXiv preprint arXiv :1912.06721

65. Risi S, Stanley KO (2019) Deep neuroevolution of recurrent and discrete world models. In: Proceedings of the genetic and evolutionary computation conference. GECCO 19. Prague, Czech Republic: Association for Computing Machinery, 456462
66. Risi S, Stanley KO (2019) Improving deep neuroevolution via deep innovation protection. arXiv:2001.01683
67. Risi S, Togelius J (2015) Neuroevolution in games: State of the art and open challenges. IEEE Transactions on Computational Intelligence and AI in Games 9(1):25–41
68. Risi S, Togelius J (2019) Procedural content generation: from automatically generating game levels to increasing generality in machine learning. arXiv:1911.13071 [cs.AI]
69. Runge J et al (2014) Chrun prediction for high-value players in casual social games. In: 2014 IEEE conference on computational intelligence and games. IEEE, pp 1–8
70. Schaeffer J et al (2007) Checkers is solved. In: Science 317.5844, pp 1518–1522. eprint: https://science.sciencemag.org/content/317/5844/1518.full.pdf
71. Schrittwieser J et al (2019) Mastering Atari, Go, Chess and Shogi by planning with a learned model. arXiv:1911.08265 [cs.LG]
72. Segler MHS (2019) World programs for model-based learning and planning in compositional state and action spaces. arXiv:1912.13007 [cs.LG]
73. Segler MH, Preuss M, Waller MP (2018) Planning chemical syntheses with deep neural networks and symbolic AI. Nature 555(7698):604
74. Shaker N, Togelius J, Nelson MJ (2016) Procedural content generation in games. Computational synthesis and creative systems. Springer, Berlin
75. Silver D et al (2018) A general reinforcement learning algorithm that masters chess, shogi, and Go through selfplay. Science 362(6419):1140–1144
76. Silver D et al (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529(7587):484–489
77. Silver D et al (2017) Mastering the game of Go without human knowledge. Nature 550:354–359
78. Spronck P et al (2006) Adaptive game AI with dynamic scripting. Mach Learn 63(3):217–248
79. Stanley KO et al (2019) Designing neural networks through neuroevolution. Nat Mach Intell 1(1):24–35
80. Such FP et al (2017) Deep neuroevolution: genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. arXiv:1712.06567
81. Summerville A et al (2018) Procedural content generation via machine learning (PCGML). IEEE Trans Games 10(3):257–270
82. Togelius J et al (2009) Super mario evolution. In: 2009 IEEE symposiumon computational intelligence and games. IEEE. pp 156–161
83. Torrado R et al (Oct. 2018) Deep reinforcement learning for general video game AI. In: Proceedings of the 2018 IEEE conference on computational intelligence and games, CIG 2018. IEEE
84. Vinyals O et al (2019) AlphaStar: mastering the real-time strategy game StarCraft II. https://deepmind.com/blog/alphastar-mastering-real-time-strategy- game-starcraft-ii/
85. Vinyals O et al (2019) Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature 575
86. Volz V (2019) Uncertainty handling in surrogate assisted optimisation of games. In: KI - Künstliche Intelligenz
87. Volz V et al (2018) Evolving mario levels in the latent space of a deep convolutional generative adversarial network. In: Proceedings of the genetic and evolutionary computation conference. ACM, pp 221–228
88. Yannakakis GN (2012) Game AI revisited. In: Proceedings of the 9th conference on computing frontiers. CF 12. Cagliari, Italy: Association for Computing Machinery, 285292
89. Yannakakis GN, Cowie R, Busso C (2018) The ordinal nature of emotions: an emerging approach. IEEE Trans Affect Comput
90. Yannakakis GN, Paiva A (2014) Emotion in games. In: Handbook on affective computing, pp 459– 471
91. Yannakakis GN, Togelius J (2015) A panorama of artificial and computational intelligence in games. IEEE Trans Comput Intellig AI Games 7(4):317–335
92. Yannakakis GN, Togelius J (2018) Artificial intelligence and games. Springer, Berlin
93. Yannakakis GN, Togelius J (2011) Experience-driven procedural content generation. IEEE Trans Affect Comput 2(3):147–161
94. Yannakakis GN, Togelius J (2011) The 2010 IEEE conference on computational intelligence and games report. IEEE Comp Int Mag 6(2):10–14
95. Yannakakis GN et al (2013) Player modeling. In: Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik
96. Zhang C et al (2018) A study on overfitting in deep reinforcement learning. arXiv:1804.06893
97. Zhu J et al (2018) Explainable AI for designers: a human-centered perspective on mixed-initiative co-creation. In: 2018 IEEE conference on computational intelligence and games (CIG). IEEE, pp 1–8