An Introduction to Game-Playing Systems and StarCraft II

Fredrik Präntare *(prantare@gmail.com)*
Linköping University

September 2, 2018

Abstract
In this document, we motivate and describe the development of artificial intelligence for game-playing. More specifically, we introduce the reader to real-time strategy games and the computer game StarCraft II as a testbed for algorithms, game-playing systems, and general problem-solving techniques.

1 Introduction

Since the inception of *artificial intelligence* (AI), game-playing has been a rewarding research endeavour, and an interesting application for a wide range of AI techniques. Today, commercial computer games are a huge part of the entertainment industry, and simulations are one of the most important tools in many fields of research.

With the emergence of cheap and powerful personal computers, simulation-based games have become extremely popular, e.g. *StarCraft* (see *Figure 1*), *Minecraft* and *Dota 2* [1,2,3,4]. Such games give rise to a plethora of interesting AI-related problems, and often induce severe time constraints that make game-playing more difficult and realistic. To be successful in these environments, players need to act and reason in real-time, be adaptable, learn to exploit opponents' mistakes, and predict the behaviour of adversaries [5, 6]. With this in mind, it is not surprising that many simulation-based games are ideal testbeds for a wide variety of real-time AI techniques, and that skills required in game-playing are central to numerous other important problems [7].

Figure 1: A screenshot from **StarCraft** (Remastered), the predecessor of *StarCraft II*.

In recent years, *real-time strategy* (RTS) games have become increasingly popular as a research environment for AI. In the next few sections, we discuss why this is the case, and why the RTS game *StarCraft II* is a very promising testbed for AI development, and the next natural step in game-playing research.

1.1 The History of Artificial Intelligence in Strategy Games

Although research in AI for strategy games and related techniques has been conducted for a long time, it was first in the early 90s that advancements in highly specialized software and hardware made it possible for computers to start challenging human experts in non-trivial strategy games. A computer that took advantage of such advancements was *Deep Blue*—a chess computer developed by *IBM* in the late 20th century. [8,9]

1.1.1 Chess

In 1997, Deep Blue managed to defeat *Garry Kasparov*, and became the world's first non-human to win an official chess match against a world champion (depicted in *Figure 2*). To achieve this level of proficiency, Deep Blue based its decision-making on simulating different moves, and evaluating subsequent game states, while utilizing *minimax*[1] and *alpha–beta pruning*[2]. [10,11]

Figure 2: A photograph from the chess match between Garry Kasparov (left) and Deep Blue.

[1]Minimax is a decision rule for minimizing the possible loss for the worst case.

[2]Alpha–beta pruning is an adversarial search algorithm. In chess, it can be used to decrease the number of states that needs to be evaluated, and thus increase the performance. Technically, it does so by pruning (discarding) branches (e.g. removing edges in a minimax tree) that cannot improve the final decision.

Today, modern chess engines (e.g. *Stockfish*, *Houdini*, *Komodo*) outperform human chess players using techniques that share similarities to those used by Deep Blue, and the seemingly ever-increasing efficiency of new computer hardware has made it possible for personal computers and chess engines to consistently outperform chess experts without difficulty. [12, 13, 14]

With Deep Blue, humankind solved the problem of creating a machine that can outperform and defeat the best human chess players. Subsequently, research in game-playing increasingly focused on more complex games, and perhaps most notably *Go*—a strategy game that has since then played an important role in game-playing research, and arguably in the history of experimental AI.

1.1.2 Go

Go (which means ”encircling game”) is a turn-based board game in which two players challenge each other in strategical reasoning and intuition. The game originates from ancient China, and is often played on a game board of 19x19 positions with playing pieces called stones. The objective of Go is to surround more territory (positions) than your opponent. A typical 19x19 Go board, with accompanying playing pieces, is shown in *Figure 3.* [15]

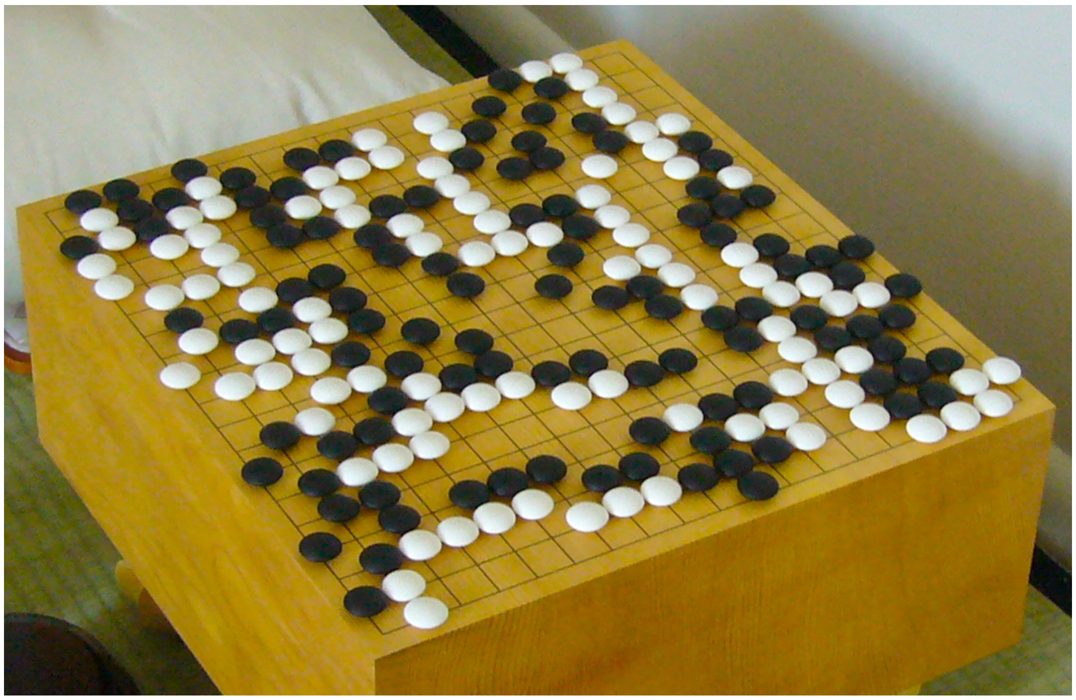

Figure 3: A typical 19x19 Go board in midst of play, with accompanying playing pieces (stones).

Even though Go has simple rules and a minimalist appearance, it is still far more complex than chess. Its state-space is huge, and it has, on average, many more possible moves per turn [16]. Also, there seems to be no obvious and efficient approach to determine whether a certain state (board configuration) is on the path to victory, making it extremely difficult to create successful Go-playing agents. With this in mind, it is not surprising that relatively straight-forward techniques, e.g. the previous search algorithms that were applied to chess, are too inefficient for Go-playing agents. Therefore, other approaches need to be considered.

Before 2015, Go-playing systems only managed to reach the skill-level of advanced amateur players [17]. In 2016, however, *AlphaGo*—an advanced Go-playing computer developed by *Google DeepMind*—became the first non-human to outperform a top-tier player in an official broadcast match (see *Figure 4*). AlphaGo managed to do so by utilizing highly sophisticated AI techniques, combined with an advanced hardware configuration consisting of more than 1000 central processing units (CPUs), 100 graphical processing units (GPUs), and the new *tensor processing unit*[3] (TPU). Its main algorithm was based on a variation of Monte Carlo tree search[4] (MCTS), guided by an advanced evaluation function that utilized artificial neural networks[5] (ANNs), and a huge database of previous Go matches. DeepMind subsequently received the inaugural *IJCAI Marvin Minsky* medal for their achievements in AI. [18, 19, 20]

Figure 4: A photograph from the famous Go match between AlphaGo and *Lee Sedol* (right).

The next year, in 2017, DeepMind published an article that introduced *AlphaGo Zero*, a new version of AlphaGo that achieved superhuman proficiency in Go through self-play and *reinforcement learning*[6] without any domain knowledge, i.e. *tabula rasa*, except for the game's rules. In three

[3]The tensor processing unit is a new type of processing unit that is specialized in tensor operations and machine learning (e.g. neural networks).

[4]Monte Carlo tree search is a search algorithm based on continuously expanding and constructing a search tree, while analysing the seemingly best moves through random playouts (i.e. simulated games).

[5]An artificial neural network is a computing system (input-output) consisting of artificial neurons that, given a certain input, changes their internal state through activation, and then produces an output.

[6]Reinforcement learning algorithms make it possible for systems to learn from their own experience, in contrast

days of training, AlphaGo Zero reached a skill-level so high that it could win against its former self, and defeat AlphaGo with a score of 100 matches to 0. A few months later, the same authors published a new article, in which they presented results from generalizing AlphaGo Zero to other games: In only 24 hours, *AlphaZero*, the generalized version of AlphaGo Zero, trained itself to a top-tier skill-level, and then convincingly defeated state-of-the-art game-playing programs in both shogi (also known as Japanese chess) and chess using the same principles of self-play. [21, 22, 23]

1.1.3 Computer Strategy Games

Even though Go is a complex game with a huge number of legal positions, the complexity of many popular computer-based strategy games overshadow the complexity of Go. Many of them, such as *Age of Empires* and StarCraft II (see *Figure 5*), not only have an immense number of possible game states, but are also partially observable (due to a *"fog of war"* that occludes what your units cannot see), highly dynamic, and played in real-time[7]. The complexity of such games make the previously mentioned approaches impractical and insufficient, making it very difficult to create game-playing agents that are on a human skill-level, even when utilizing highly specialized hardware and software.

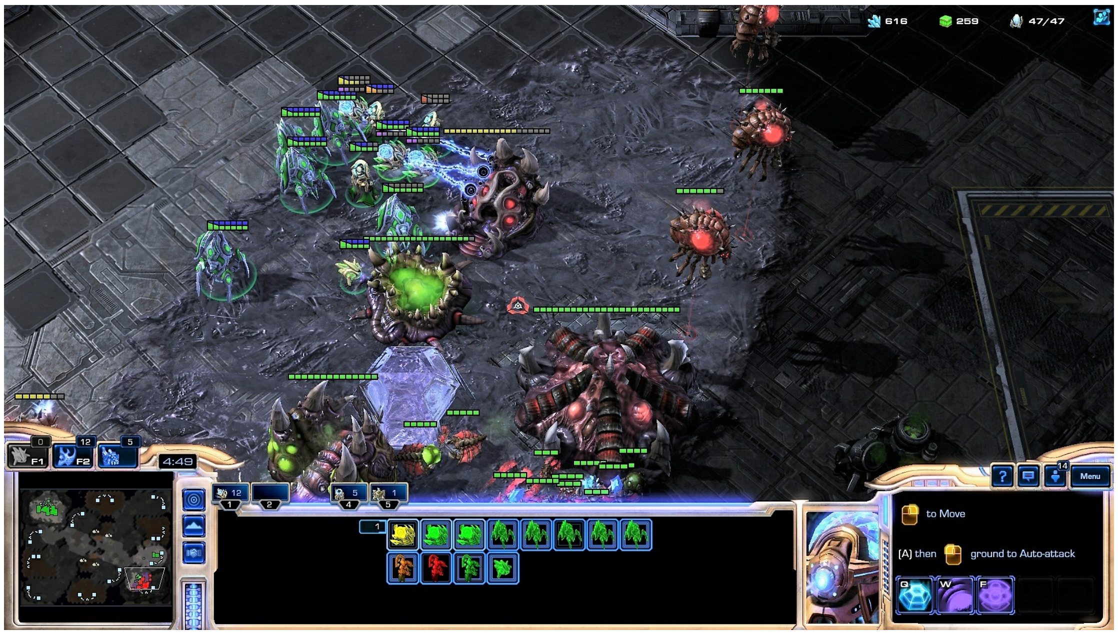

Figure 5: An image of a typical scenario in the strategy game StarCraft II, where the red player is on the verge of defeat due to not having reacted appropriately to an aggressive strategy (known as "cheese") executed by a Protoss (one of the three playable races in the StarCraft series).

With DeepMind's extensive advancements in game-playing, researchers and AI experts increasingly turned their attention to more complex simulation environments that are more similar to the

to learning from supervision-mechanisms, such as expert knowledge, or labelled data sets.

[7]A real-time game is a game in which all players can act simultaneously at any given time.

real world. For example, DeepMind worked together with *Blizzard Entertainment*[8], and released the RTS game StarCraft II as a simulation environment for AI research in late 2017. To motivate experimental research in StarCraft II, *Oriol Vinyals*, a deep learning scientist at DeepMind, wrote:

> *"StarCraft is an interesting testing environment for current AI research because it provides a useful bridge to the messiness of the real-world. The skills required for an agent to progress through the environment and play StarCraft well could ultimately transfer to real-world tasks."*

Additionally, a research team at *Facebook AI Research* published a paper on using machine learning for handling unit micromanagement in StarCraft. They also developed *TorchCraft*, a programming library that is focused on enabling deep learning research for RTS games. [4,24,25]

OPENAI FIVE BENCHMARK
OpenAI 4 (Bot)
OpenAI 4 (Bot): We estimate the probability of winning to be above 90%.
EARTHSHAKER
OpenAI

Figure 6: A Dota 2 screenshot from the OpenAI Five match that took place in August 2018.

Apart from StarCraft, there has also been work in developing bots for the game Dota 2. In fact, Dota 2 is the only non-trivial strategical real-time game for which game-playing agents have been able to defeat human players in an official match. This was accomplished by *OpenAI Five*—a team of self-learned Dota 2 bots—when they defeated a non-professional team of well-known humans in a series of matches in August 2018, depicted in *Figure 6* (however, a few days later, they were crushingly defeated by a professional team). Similarly to AlphaZero, OpenAI Five utilised learning via self-play. The authors described this self-learning process, which ran on 256 GPUs and 128000 CPU cores, as having a bot play "180 years worth of games against itself every day", and explained

[8]Blizzard Entertainment is the creator of the StarCraft game series.

that it consists of a *long short term memory*[9] (LSTM) network to take dynamic behaviour into consideration. [1]

In Dota 2, the player only controls one unit (the hero), but to win, she has to work together with her team of 4 other players to defeat the enemy team. In StarCraft II, the player instead has to produce her own units (up to many hundreds per match), and coordinate (micromanage) each of them individually. With this in mind: In the following section, we give an in-depth description of StarCraft II, and its potential as a simulation environment for the development of more general game-playing methods that could ultimately transfer to the real world.

1.2 StarCraft II

StarCraft II is a military RTS game set in a science fiction setting with three asymmetrical playable races (the *Zerg*, *Protoss*, and *Terran*—depicted in *Figure 7*) released in 2010. It has a large community, with many professional players that compete to win large prize pools in frequent competitions, such as the *World Championship Series*, seen in *Figure 8*, and the *Global StarCraft II League* [26].

Figure 7: A depiction of the three asymmetrical races available in StarCraft II: the Zerg (left), Protoss (center) and Terran (right). Each race has its own unique abilities, units, and potential strategies.

Figure 8: A photograph from the finals of the StarCraft II World Championship Series (2014).

[9]The long short term memory network is a variation of the *recurrent neural network*—a type of ANN that can use its internal units to process sequences of inputs, specifically designed to handle tasks that require taking temporal behaviour into consideration.

The goal in StarCraft II is to defeat all opponents by destroying all their buildings (but matches almost always end in the voluntary resignation of the losing players). To accomplish this goal, players have to make quick decisions in regards to infrastructure, units, tactics, and resources. The screenshot in *Figure 9* shows some of these processes—resource gathering, production, and base-building—from a Terran player's perspective.

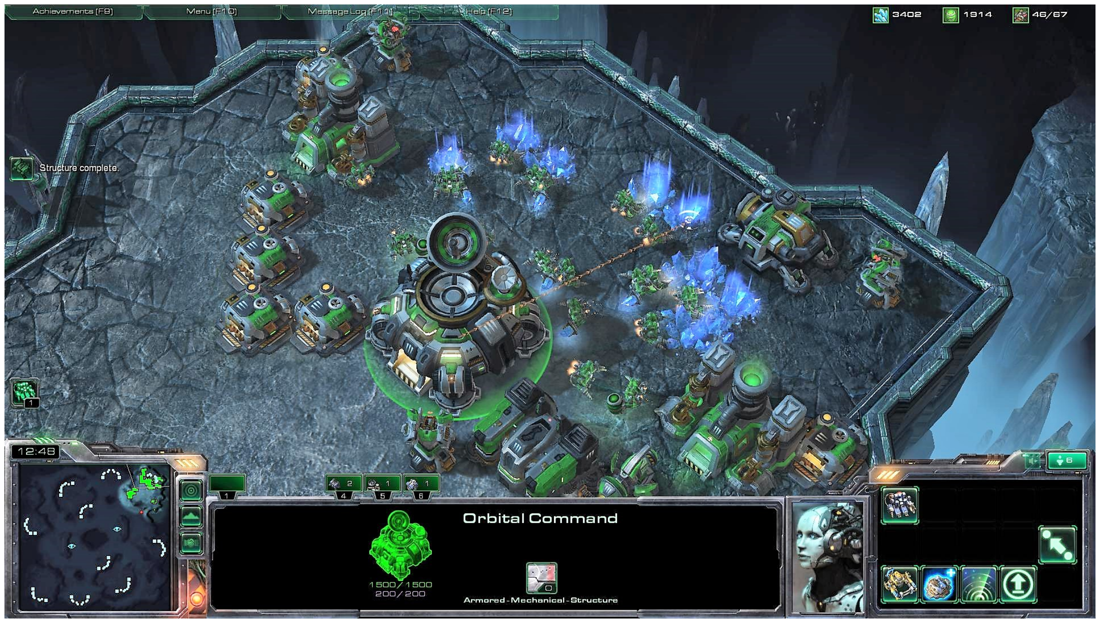

Figure 9: A screenshot from a player's perspective in StarCraft II. The screenshot shows an early-game Terran base, with several SCVs (workers) that are collecting minerals (the blue crystals) and vespene gas—the two resource types that are available in the game.

Matches in StarCraft II are mostly played one-against-one on predefined maps. These maps are to a large degree static, but very different from each other, and their decompositions continuously force the players to take the terrain and its attributes into consideration. See *Figure 10* for examples.

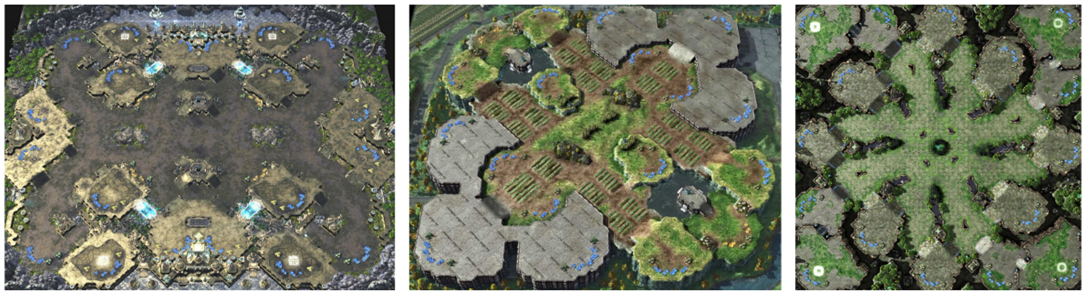

Figure 10: A bird's eye view of three different maps from StarCraft II. The blue arcs are minerals, and they indicate potential base locations.

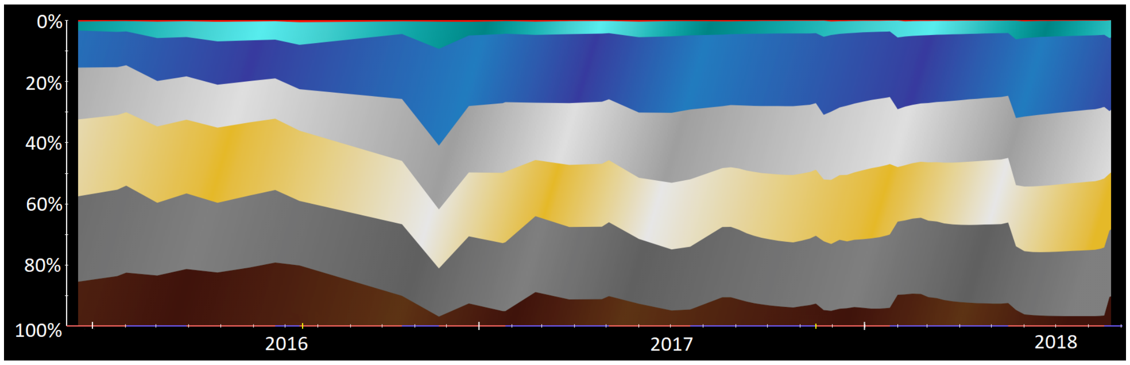

Figure 11: The 1v1 league distribution over time for StarCraft II players. From top to bottom: The Grandmaster (red—a tiny line on the top of the graph), Master (teal), Diamond (blue), Platinum, Gold, Silver, and Bronze leagues, respectively. *Courtesy of rankedftw.com.*

At the start of a match, every player is given a main base, and 12 workers (Zerg players are also given an Overlord unit—a flying unit that provides supply and vision). Using these units, the players research technology (upgrades), construct new buildings, produce new units, wage war against enemies (e.g. attack, defend), and expand their territory. To maximise efficiency, professional players make over 300 *actions per minute* (APM). A single action in StarCraft II may drastically change the outcome of a match, and to master the game, players have to react as fast as possible to the actions and strategies of their adversaries. Even top-tier amateur players, such as non-professional *Grandmaster*[10] (see *Figure 11* for a visualisation of the 1v1 league distributions over time for StarCraft II players) players, need to have almost perfect high-precision control. Important decisions, ideas, and reactions, e.g. those that can change the outcome of a match, are often discussed and analysed in terms of certain recurring concepts, including:

- ***Build Order***: A pre-defined production progression (analogous to openings in e.g. chess).
- ***Micromanagement* and *Macromanagement***: The management of units and production (technology included), respectively.
- ***Strategy***: A grand plan of action that often dictates how to react to different hypothetical build orders of the opponent.

Note that these concepts are intertwined. For example, build orders and macromanagement are almost always affected by strategy, and they, in turn, alter the possibilities for micromanagement.

Players also have to consider *meta-strategies*, and top-tier players often base their strategies on explicit assumptions of their enemies' preconditions and preferences (e.g. preferred build orders) to hard-counter them. This is an important consideration in StarCraft II due to the players' incomplete information of the game state. By making correct assumptions of opponents, players can optimise their play through ”short-circuiting”, e.g. by saving time and resources after skipping tasks that they would normally have to attend to. The dynamics that describe this type of behaviour are

[10]Players in StarCraft II are ranked based on their *MMR*, a rating that indicates the relative skill of each player. Players are placed in one out of 7 different leagues based on this rating: Bronze, Silver, Gold, Platinum, Diamond, Master and Grandmaster. The Grandmaster league contains the top 1000 best players in the world.

known as the *meta-game*, which refers to the preparations and thought-processes of players that they do before matches to gain potential advantages. This is an important concept for StarCraft II players, since the expected win rate of a strategy often depends, to a large degree, on the player that they are playing against. For example, professional players thoroughly study their opponents before matches, analyse the latest strategy-trends, and take advantage of psychological warfare to exploit opponents' mental states and strategies: If a player loses to a strategy that they haven't played against before, he or she may, in a succeeding match, play overly safe to prevent losing from the same strategy again, consequently becoming weaker against other strategies. A famous example of a player that often takes advantage of such tactics is the arguably cunning *Kim "sOs" Yoo-jin* (see *Figure 12*)—one of the highest ranked players in StarCraft II's relatively short history—who is well-known for continuously inventing new strategies and build orders to surprise his enemies through a deliberate meta-game style of play.

Figure 12: The professional StarCraft II player Kim "sOs" Yoo-jin before one of his matches in Proleague (2015). He is well-known in the game's community for his involvement in continuously altering the meta-game and persistently inventing new strong strategies.

2 Artificial Intelligence and StarCraft II

With the previous sections in mind, it is not surprising that StarCraft II is a very complex game with an ever-changing meta-game. There exists some tools that can help the players and bots make decisions and preparations to decrease their workload, e.g. databases of recorded games (replays), and also in-game assistance, including, but not limited to:

- ***Pathfinding***: When a unit is ordered to move to a position, it will find and follow the shortest path to it without the player's interaction.
- ***Resource Gathering***: When a worker has been ordered to gather resources, it will continue to do so automatically until ordered otherwise.

- ***Attack Move***: If a unit is ordered to "attack move" a position, it will try to move there, and attack all units that it encounters on its path.

With this in mind, and even though it is relatively simple to create a basic game-playing bot that can defeat amateurs, researchers and AI experts have thus far failed to develop and present a game-playing agent with a skill level that is on par with top-tier human players. This is partly due to StarCraft II's immense state-space (see *Figure 13*), and the multi-agent nature of its game mechanics—coordinating multiple units perfectly can be difficult, especially in complex environments when taking long term strategies and learning (adaptation) into consideration. However, these are also some of the main reasons to why StarCraft II is a promising testbed for AI algorithms and game-playing techniques, since the game inherently offers a dynamic multifaceted simulation with a wide range of interesting problems to solve, in a domain that is both intuitive and, in many aspects, more real-world-like than many other games.

Game	**Branching Factor**	**Average Depth**	**Possible States**
Chess	35	80	10^{46}
Go	$[30, 300]$	$[150, 200]$	10^{170}
StarCraft	$[10^{50}, 10^{200}]$	36000	10^{1685}

Figure 13: Approximate branching factors, average depths, and sizes of state spaces for chess, Go, and StarCraft: Brood War [5, 16, 24].

Instead of attempting to use a single holistic technique (e.g. those that were successful in chess) for StarCraft II, many researchers and AI experts instead focus on combining different techniques to create more efficient bots. To do so, one may divide game-playing into more manageable problem areas that can be solved independently. For example, in 2003, Buro identified six such fundamental AI research areas in RTS games:

- **Resource Management**.
- **Decision-Making under Uncertainty**.
- **Spatial and Temporal Reasoning**.
- **Collaboration**.
- **Opponent Modelling and Learning**.
- **Adversarial Real-Time Planning**.

These six areas are intertwined, and have been subject to substantial effort since then. There are many possible approaches to all of them, each with their own strengths and weaknesses. We now describe how these problem areas relate to StarCraft II in the following subsections, before we conclude with a summary. [5, 7, 27, 28]

2.1 Resource Management

Units and buildings cost resources to produce. The two available resource types in StarCraft II are minerals and gas. Without efficient management of these, a player may fail to keep production on par with adversaries (see *Figure 14*), and ultimately lose the game. Solving this problem typically involves estimating future availability of resources, so that the player can plan ahead to build new mining bases before old ones are depleted, and optimising resource gathering through planning (strategies) and low-level control (micromanagement).

Figure 14: Resource management is crucial to sustain a competitive level of production. This screenshot shows a Protoss player's mining base, and several stargates (the air-production facility of the Protoss) that are producing carriers—one of the most expensive units in the game.

2.2 Decision-Making under Uncertainty

StarCraft II is a partially observable game, and players have to deal with uncertainty due to the game's *fog of war* (depicted in *Figure 15*). In other words, the players have to act on imperfect observations with unknown outcomes. They may, for example, have to estimate threat levels, and navigate unexplored environments. To handle these types of problems, a bot may have to take advantage of probabilistic models, such as *Markov decision processes* (MDPs) to model sequential problems, or *Bayesian networks* to describe the probabilistic relationships of variables.

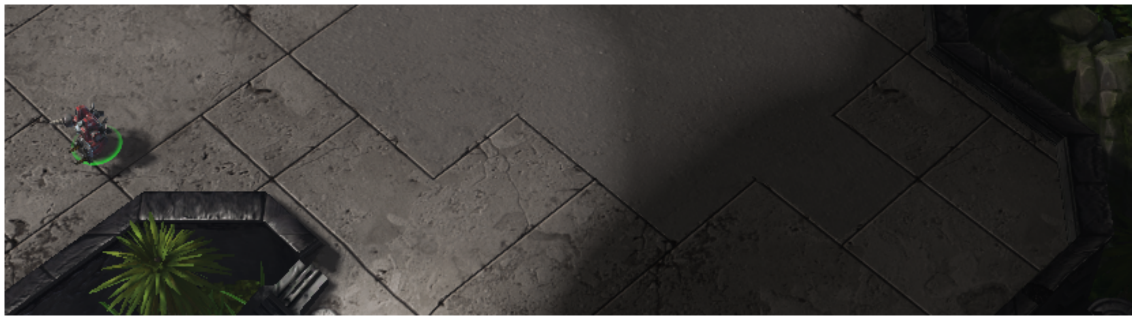

Figure 15: The fog of war (the darker area to the right in this image) conceals areas that are not in range of friendly units, thus forcing players to make decisions under uncertainty.

2.3 Spatial and Temporal Reasoning

A strategy in StarCraft II is typically only viable in certain time frames and environments, and geospatial information, such as cliffs and destructible objects, can often be taken advantage of to make better decisions (see *Figure 16*). Spatial and temporal reasoning involves applying logic to solve spatio-temporal problems that relate to such data—for example by utilizing *path planning* to steer units, or generating plans for optimal production and the development of new upgrades.

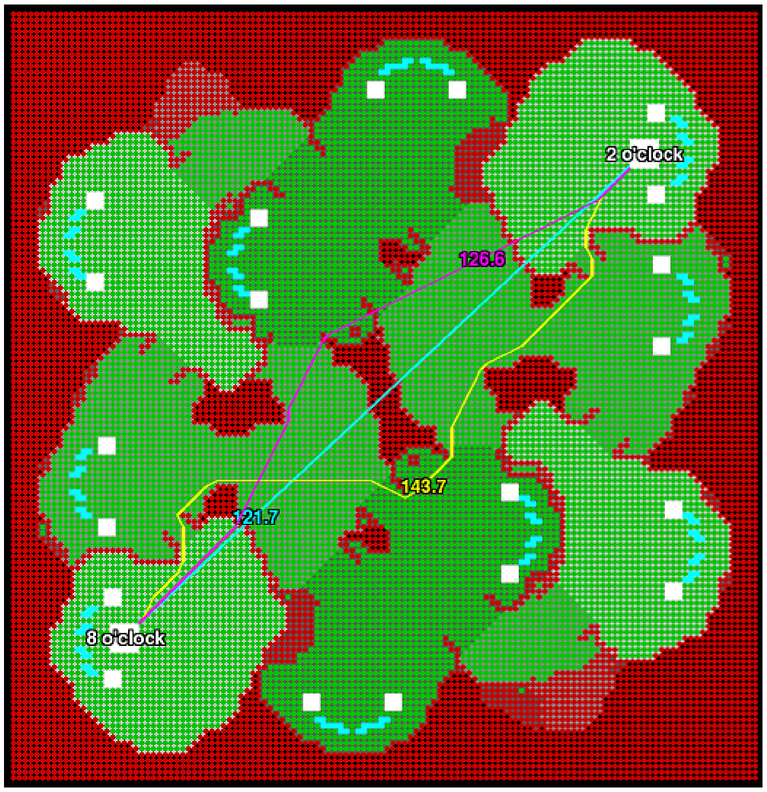

Figure 16: Movement plans can make it possible for players to make better decisions, thus improving their game-playing skills. This image shows three different types of shortest paths between two base locations: By air (teal), by ground (yellow), and by cliff-walking (purple). The paths were generated by the *SC2 Map Analyzer*.

2.4 Collaboration

To succeed in playing StarCraft II, the player has to command many units simultaneously. From another perspective: The player has to coordinate and organize multiple units so that the units can work together to solve tasks, reach goals, and improve their problem-solving skills. This makes it possible for a single unit to transcend its own capabilities via cooperation with other friendly units. There are many organizational paradigms that can be utilized for this purpose (e.g. *coalitions* and *hierarchies*), and there are algorithms for *task allocation* that can be used to coordinate units.

2.5 Opponent Modelling and Learning

Players in StarCraft II can potentially improve their game-playing skills by learning how to exploit the mistakes of their opponents. Perhaps more importantly, behaviours can be optimised by learning from one's own mistakes, and make players more robust to unexpected strategies and unconventional tactics, e.g. by extrapolating experience. For example, online reinforcement learning can be used to continuously improve unit-control, and *supervised learning* can be used to generate player models, e.g. by analysing replay data, to predict strategies and behaviours of adversaries.

2.6 Adversarial Real-Time Planning

In chess and Go, search techniques (e.g. Monte Carlo tree search) have been used to solve problems that are inherently adversarial. In StarCraft II, such search techniques are typically too inefficient for direct use. By utilizing high-level abstractions that take opponents' potential strategies into consideration, adversarial real-time planning can be used to generate strategies, while playing, that are less vulnerable to hostile interdiction.

3 Conclusion

In this document, we introduced the reader to strategy games and game-playing systems. More specifically, we introduced the reader to RTS games and StarCraft II, and their utility as testbeds for benchmarking and developing algorithms. We deliberately emphasized StarCraft II for this purpose, since it is arguably one of the most influential games in the AI community right now. It is also very multifaceted (depicted in *Figure 17*), while still being more similar to the real world compared to other well-studied games (e.g. poker, chess and Go). It is now open to the public, and also cost-free, making it is easier than ever to develop new game-playing bots.

Figure 17: StarCraft II is a multifaceted game with a wide range of aspects that players have to consider, and it offers a dynamic real-time environment that poses many algorithmic challenges.

References

[1] OpenAI, “Openai five,” 2017.

[2] K. Aluru, S. Tellex, J. Oberlin, and J. MacGlashan, “Minecraft as an experimental world for ai in robotics,” in *AAAI Fall Symposium*, 2015.

[3] A. Drachen, M. Yancey, J. Maguire, D. Chu, I. Y. Wang, T. Mahlmann, M. Schubert, and D. Klabajan, “Skill-based differences in spatio-temporal team behaviour in defence of the ancients 2 (dota 2),” in *Games media entertainment (GEM), 2014 IEEE*, pp. 1–8, IEEE, 2014.

[4] O. V. (DeepMind), “Deepmind and blizzard to release starcraft ii as an ai research environment,” 2016.

[5] S. Ontanón, G. Synnaeve, A. Uriarte, F. Richoux, D. Churchill, and M. Preuss, “A survey of real-time strategy game ai research and competition in starcraft,” *IEEE Transactions on Computational Intelligence and AI in games*, vol. 5, no. 4, pp. 293–311, 2013.

[6] M. Buro and T. M. Furtak, “Rts games and real-time ai research,” in *Proceedings of the Behavior Representation in Modeling and Simulation Conference (BRIMS)*, vol. 6370, 2004.

[7] M. Buro, “Real-time strategy games: A new ai research challenge,” in *IJCAI*, pp. 1534–1535, 2003.

[8] IBM, “Ibm100 - deep blue,” 2017.

[9] F.-h. Hsu, “Ibm’s deep blue chess grandmaster chips,” *IEEE Micro*, vol. 19, no. 2, pp. 70–81, 1999.

[10] M. Newborn, *Kasparov versus Deep Blue: Computer chess comes of age.* Springer Science & Business Media, 2012.

[11] M. Campbell, A. J. Hoane, and F.-h. Hsu, “Deep blue,” *Artificial intelligence*, vol. 134, no. 1-2, pp. 57–83, 2002.

[12] S. Developers, “Stockfish source code,” 2017.

[13] R. Houdart, “Houdini chess engine website,” 2017.

[14] P. Banjan and A. Silver, “Komodo: Birth of a chess engine,” 2016.

[15] A. G. Association, “A brief history of go,” 2017.

[16] J. Tromp and G. Farnebäck, “Combinatorics of go,” in *International Conference on Computers and Games*, pp. 84–99, Springer, 2006.

[17] N. Wedd, “Human-computer go challenges,” 2018.

[18] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[19] G. DeepMind, “Alphago,” 2016.

[20] IJCAI, “Marvin minsky medal for outstanding achievements in ai,” 2017.

[21] D. Hassabis and D. Silver, “Alphago zero: Learning from scratch,” 2017.

[22] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, “Mastering the game of go without human knowledge,” *Nature*, vol. 550, no. 7676, p. 354, 2017.

[23] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, “Mastering chess and shogi by self-play with a general reinforcement learning algorithm,” *arXiv preprint arXiv:1712.01815*, 2017.

[24] N. Usunier, G. Synnaeve, Z. Lin, and S. Chintala, “Episodic exploration for deep deterministic policies: An application to starcraft micromanagement tasks,” *arXiv preprint arXiv:1609.02993*, 2016.

[25] G. Synnaeve, N. Nardelli, A. Auvolat, S. Chintala, T. Lacroix, Z. Lin, F. Richoux, and N. Usunier, “Torchcraft: a library for machine learning research on real-time strategy games,” *arXiv preprint arXiv:1611.00625*, 2016.

[26] B. Entertainment, “Starcraft ii world championship series (wcs),” 2018.

[27] S. Ontanón, K. Mishra, N. Sugandh, and A. Ram, “On-line case-based planning,” *Computational Intelligence*, vol. 26, no. 1, pp. 84–119, 2010.

[28] D. W. Aha, M. Molineaux, and M. Ponsen, “Learning to win: Case-based plan selection in a real-time strategy game,” in *International Conference on Case-Based Reasoning*, pp. 5–20, Springer, 2005.