Chess Engines and Quantum Mechanics

Research Report

Peter Murphy
16440004

COMP47340: Computational Thinking

UCD School of Computer Science

DATE: 7/12/2020

Topic 1: Chess Engines

On slide 11 of the talk *Computers Change the Way We Solve Problems*, Padraig mentioned Deep Blue, a chess engine that used a brute-force approach to defeat the world champion, Garry Kasparov, in 1997. As chess is something I'm quite interested in, I wanted to learn more about the history of chess engines.

The First Chess Engines

Shortly after WWII, the "father of computer science", Alan Turing, began writing the code for one of the first ever chess engines, which he called "Turbochamp", before computers could even execute such code. In 1952, after failing to implement his program on the Ferranti Mark-I, one of the world's first commercially available digital computers, Turing challenged his colleague Alick Glennie to a game of chess where he implemented his code with pencil and paper, taking over 30 minutes to calculate each move. Turbochamp ultimately lost the game in 29 moves, but this was arguably the first ever full chess game played by a "computer".

During this same period, Claude Shannon, who was also laying the groundwork for the digital age, was interested with the idea of a chess-playing computer. In 1950, he published one of his many ground-breaking papers, *Programming a Computer For Playing Chess*, where he pointed out that solving this problem differed from the general use of computers at that time in some key ways, the inputs are not simply numbers but chess positions, finding the best moves requires some type of judgement by the computer and the output won't be right or wrong, but somewhere in a continuous range of quality from "bad" to "good". He describes the mathematics for two approaches to solving this problem, Type A and Type B. In principle, all chess positions fall into one of three categories, won, drawn or lost, as the game must end after a finite number of moves (if no pieces are taken after a series of 50 moves, the game is drawn). An evaluation function takes a position as an input, and calculates which category the position falls into. Therefore a computer can theoretically calculate all the possible moves from an initial position, use this evaluation function on every position and working backwards from the end of every possible combination, it could determine, with perfect play from both sides, if the position is won, drawn or lost. However this isn't practical, as the number of combinations is too high, with a rough estimate of 10^{120} combinations. A basic evaluation function can be constructed using factors like, the value of the pieces, (1 for pawns, 3 for knights and bishops, 5 for rooks, 9 for queens), the number of doubled, isolated and backwards pawns and the number of possible moves. The computer could use this evaluation function on a position to estimate how good the position is, instead of determining exactly if it's won, drawn or lost. It could then use this function to calculate how good all the positions from all possible response moves are, and then continue this idea for a number of further response moves, where increasing the depth would of course increase the quality of move selection but also the time taken to calculate. This type of approach is what Shannon called Type A.

To improve on this method, the computer needs some sort of strategy for choosing which moves to consider and when to stop calculating each variation,

such as examining only relatively stable positions or calculating forced variations as far as possible, which Shannon called Type B.
Both Turing and Shannon, 2 of the most important minds in the history of computer science, saw the value in solving this problem, not in the intrinsic value of a computer that can win a chess game, but how it can win. It is generally believed that achieving a high skill level in chess requires some sort of intelligence, and as the problem is not so easy that it's trivial nor so hard that it's impossible, this would be a good problem for developing artificial intelligence, though this term was not used at the time.
In 1957, an IBM engineer, Alex Bernstein, created the first automated program capable of playing a full game of chess, and so began the rise of the chess engines.

Surpassing Humans

In the 60s, the MiniMax algorithm developed by John Von Neumann set the foundation for vast improvements in chess engine algorithms, and as the years went by, computing power increased exponentially and so chess engine progress steadily increased year after year. In 1967, MacHack Vi became the first engine to beat a human, and by the end of the 70s, engines could easily beat low to mid level players, but were still no match for masters.
Then in the 1980s, as computers were becoming much more widespread and the chess engine industry was becoming very lucrative, engines finally began beating the top players. However, in 1985, the world champion Garry Kasparov simultaneously defeated 32 of the world's top chess engines in an incredible exhibition in Hamburg. He also defeated Deep Thought, the first engine to reach Grand Master level, in 1989, but it was clear the computers would soon take over. IBM acquired Deep Thought, which they renamed to Deep Blue, and 8 years later, we arrive at the famous 1997 match between Deep Blue and Garry Kasparov, where for the first time, the reigning Chess World Champion was defeated by a computer. During the next few years, humans made their last stand against the machines, refusing to go down without a fight, until Hydra started destroying Grand Masters in 2004 and Deep Fritz beat the World Champion Vladimir Kramnik in 2006.

AlphaZero

As chess engines were now clearly superior to humans, they became essential for all tournament players and continued their steady growth for the next 10 years without any major changes, until December 2017. At this point, all engines were essentially of the Type B that Shannon described, with Stockfish the highest rated in the world.
DeepMind offered a completely new approach to the problem by using neural networks, and in doing so took a huge step forward in the development of artificial intelligence. After AlphaGo defeated Lee Sedol in 2016, as mentioned in the lecture, DeepMind developed the more general algorithm AlphaZero. which they then applied to the games of Go, chess and shogi. With nothing but the rules of chess as input, after 4 hours of learning by playing against itself, AlphaZero surpassed the level of Stockfish, crushing Stockfish with 28 wins, 72 draws and 0 losses in a 100 game match. There was criticism of the circumstances under which these games were played, but they were dispelled a year later when a second paper was published with further detail. Previous engines had no

"understanding" of chess, the principles of chess were understood by humans, such as the value of each piece, the safety of the king, control of the center, etc., and had to be implemented in the evaluation function, which caused engines to play a distinctly non-human, brutish style of chess, as they had no insight into these principles. What was truly remarkable about AlphaZero was how it played, with a style unlike any engine before it. It rediscovered these principles on its own and even improved on them, and seemed to play with a human-like insight, such as taking risks, like sacrificing pieces, to gain an advantage. It calculated about 1000 times less positions per second than Stockfish, and even when Stockfish was given ten times as much time per move, it was still inferior. As Kasparov mentioned in a commentary accompanying the paper, AlphaZero works smarter, not harder than other engines, and as it forms its own principles free from the bias of programmers, it's style reflects some fundamental "truth" of chess.

Why Is This Important?

The ultimate goal of all this, as Shannon pointed out back in 1950, was not to build programs to win chess games, but to use what we learn through all this research and development to discover similar truths of things that are actually important, and DeepMind have already begun doing this, with incredible results. They developed AlphaFold, a program that uses deep-learning to predict the structure of folded proteins to within the width of an atom, a problem that has plagued biologists for over 50 years. In the 1960s, researchers found that if they could calculate all the interactions within a protein's sequence of amino acids, they could predict it's 3D shape, however this method is very computationally expensive. In 1994, a world-wide experiment/competition called the Critical Assessment of Protein Structure Prediction (CASP) began, and has taken place every 2 years since. It gives research groups a chance to test their protein prediction methods against experimental results and compete with other research groups. In 2018, AlphaFold beat the competition easily in it's first appearance, with an accuracy of around 60% on the hardest tests, but John Jumper, the lead on AlphaFold's development, said this was still too low for practical use. In preparation for the 2020 CASP, AlphaFold was trained using the 170,000 known protein structures (there are roughly 200 million proteins in total), and on the 30th November 2020, it was reported that AlphaFold achieved an accuracy of 87% on the hardest proteins, 25% higher than second place, becoming the first theoretical prediction to match the accuracy of experimental methods. John Moult, who heads the team that founded and runs CASP, said "It's the first use of AI to solve a serious problem". AlphaFold can do in a few days what usually takes hundreds of thousands of dollars and years of experiments. To make sure that AlphaFold didn't cheat, Andrei Lupas, one of the CASP judges, set a challenge, to model a membrane protein from a species of archaea, an ancient group of microbes, something his research team failed to do for 10 years. Of course, AlphaFold returned an accurate model, which allowed Lupas to make sense of his experimental results, and after fitting his results to AlphaFold's model, he said, "It's almost perfect, I don't know how they do it". AlphaFold did noticeably fail on one protein, which Jumper said would be the next problem they tackle, following this remarkable result.
Wordcount: 1709

Topic 2: Quantum Computers

In *Computers Change the Way we Solve Problems*, Padraig talked about how computers solve problems. During my undergrad in physics, I remember learning that quantum computers had the potential to solve problems that classical computers couldn't, so I decided to research quantum computing.

The Beginning

In the 1960s, as the electronic components of computers were reaching smaller scales every year, Richard Feynman first proposed the idea to exploit the quantum nature of these components to increase the performance of computers, in particular, using the concepts of superposition and entanglement. In a classical computer, a bit represents a physical state in the computer's memory, for example, a transistor represents a 1 when it's turned on or a 0 when it's turned off.

In quantum mechanics, states are not as clearly defined. The state of a system exists in a superposition between all possible states until the system is measured, which causes the system to collapse into one of the possible states. This is represented mathematically in a wave function, where the amplitude associated with each state is the probability that the system will collapse into that state. This can be demonstrated in Young's dual slits experiment, where a photon appears to pass through two slits at the same time and form an interference pattern with itself, until you "look" at one of the slits, i.e. measure the wave function and therefore collapse the system into one of the possible states, and then the electron only passes through one slit and there's no interference pattern.

Entanglement describes how the states of 2 or more particles become linked after interacting in a specific way.

In 1985, David Deutsch published a paper describing how to build a quantum logic gate, and 10 years later in 1995, Christopher Monroe and David Wineland, did just that. They used the energy levels of an ion to create the first ever quantum bit, or qubit, where a lower energy level represented a 0, and a higher energy level represented a 1. They then added a second qubit to the system, by measuring the external motion of the ion, where low motion represented a 0 and high motion represented a 1, and combining these 2 qubits formed a quantum version of the CONTROLLED NOT gate. This idea forms the building blocks for quantum computers. As the qubits exist in a superposition between 0 and 1, they can perform many calculations simultaneously, which would increase computing power exponentially, and would be extremely useful for a number of real world problems, like the travelling salesman problem mentioned in the lectures.

Also in the 1990s, Peter Shor was tackling the problem of quantum error correction. Classical computers use the method of redundancy for error correction, where a number of copies of the information are stored, and if any of these copies are found to disagree due to some error, they cross-reference with each other and remove the minority. However, due to the no-cloning theorem, quantum states can't be copied, and due to their extremely delicate nature, it is very important to have viable error correction. Shor made a breakthrough in 1995, when he published a paper describing how to redundantly encode quantum information using an expanded system of entangled qubits, and with subsequent developments on his work, quantum error correction became plausible, turning practical quantum computing into a real possibility, though it still remains the biggest obstacle in increasing the number of qubits. Shor also devised an algorithm for a quantum computer to find the prime factors of an integer, which forms the basis of most modern encryption methods, along with the discrete logarithm and elliptic-curve discrete logarithm encryptions. A sufficiently powerful quantum computer would theoretically be able to break such encryptions relatively easily, meaning quantum-proof encryption methods will need to be created.

Current Quantum Computers

There are a number of ways to store quantum states, such as the energy levels of ions as mentioned above, or the spins of atoms or polarization of light, but the most popular method used at the moment, by companies like Google and IBM, is to cool silicon tubes to nearly 0 Kelvin, to create superconductors that heighten the effect of quantum mechanics. In 2019, Google revealed it's Sycamore chip, a programmable quantum computer with 54 qubits, where each qubit is connected to 4 other qubits, and claimed to have achieved 'quantum supremacy', a long-awaited milestone, where a quantum computer performs a task that is practically impossible for a classical computer. They performed a complex calculation in 200 seconds that they estimated would take the world's most powerful super-computers 10,000 years, though this claim was disputed by IBM, claiming their supercomputer, Summit, could do it in 2.5 days. The result was still significant, as it offered the first evidence that quantum supremacy is practically achievable.

Then in September 2020, using 12 qubits, Sycamore successfully simulated a chemical reaction, known as the Hartree-Fock procedure, however this is a trivial problem for modern computers. The point was to prove that the computer is programmable, and that it can solve important problems in the future, when hardware and software are substantially improved.

China has also made quantum technologies a high priority in recent years, investing $10 billion in the largest quantum research facility in the world in 2017. On December 3rd 2020, a team in China claimed to have achieved the first example of definitive quantum supremacy. They stored the qubits using the polarization of photons, which they used to perform Boson sampling in a few minutes, a calculation that has been proven to be impossible on classical computers as it is a #P-hard problem, and which has possible practical applications in machine learning, quantum chemistry and graph theory. However, their 'computer' isn't programmable, and was designed only for this problem, but still remains an important breakthrough.

The future of quantum computing is very exciting as the possibilities are endless, with the potential to revolutionize fields like medicine, communication and all the sciences. It is the natural next step in the evolution of computers, but the physical and engineering challenges mean it could be quite some time before we reach them.

Wordcount: 1026

Topic 3: Cryptography

On slide 9 of the talk *Poems That Solve Problems*, Chris talked about the Enigma code used during WWII, and how Turing helped design the bombe to break the encryption used by the Germans. So following on from quantum computing, I decided to research quantum cryptography, as I find quantum technology really interesting.

Classical Cryptography

Cryptography is the practice of creating techniques to secure information for communication, and is thousands of years old. These techniques usually involve coding the information to be sent, called plaintext, into cyphertext, which is then deciphered by the recipient using a key, which both parties must agree on. As the key must be kept secret from attackers, it needs to be distributed via a 'secure channel'.

In the 1940s, Claude Shannon published a number of groundbreaking papers on the mathematics of communication and cryptography. His papers formed the foundation of information theory and modern cryptography. He identified the two main goals of cryptography, secrecy and authenticity, and described the two general methods to ensure secrecy, one to defend against attackers with unlimited resources, called theoretical security, and one to defend against attackers with finite resources, called computational security.

As technology became more widespread in the following decades, improvements in cryptography were desperately needed, and in 1976, a paper published by Whitfield Diffie and Martin Hellman, titled "New Directions in Cryptography", provided a solution to the problem of key distribution. They described the public key or asymmetric key system, where two connected keys are produced that can decode messages encrypted using the other key, and one key cannot be found using the other key. This allows a user to have a public key, which other users use to encrypt messages for them, and a private key, which they use to decrypt these messages. The security of these keys relies on computationally expensive mathematical problems, such as finding the prime factors of a large number, but this means they require longer keys to match the security of symmetric keys, so a hybrid system is usually used, where an asymmetric key is used to exchange a symmetric key between two parties, which they then use for communication.

The following year, Ron Rivest, Adi Shamir and Leonard Adleman implemented this method in their RSA cryptosystem, which remains one of the most used systems in the world today, with RSA keys in the range of 1024 to 4096 bits.

As I mentioned in the previous section, a powerful enough quantum computer could easily break these encryptions. In 2012, a 4 qubit computer factored 143, and in 2014, it factored 56,153. Recent improvements to Shor's algorithm have shown a 20 million qubit computer could break a 2048 bit RSA encryption, and while such a computer won't be coming anytime soon, researchers believe it to be mainly an engineering challenge, and that these computers may arrive 20 to 30 years from now. This threat has lead to the development of quantum-proof methods, such as lattice-based cryptography, but so far, none of these methods seem promising enough to replace what is currently in use, and further research is urgently needed.

Quantum Cryptography

The one-time pad is the only classical encryption method known to be theoretically secure, where a one-time key is used. The key must be as long as the plaintext, randomly generated, never reused and kept secret, and it has been proven that any perfectly secret system must use a key with essentially the same properties, but as this requires a secure channel to send the key, it is not very useful in practice.

Quantum cryptography offers an alternative for theoretically secure systems, as it relies on quantum mechanics instead of complex mathematical problems. Lots of research has been done on quantum key distribution (QKD) in particular, where bits that form a key are transferred using randomly polarized photons. An attacker can't "look" at the key, as measuring a quantum state disturbs it, and they can't copy the key, due to the no-cloning theorem, so this allows the exchange of truly secret, random keys.

While China is lagging behind America in developing a quantum computer, they are leading the way in quantum cryptography, and with the launch of Micius in 2016, a satellite dedicated to quantum communication, a team led by Pan Jian-Wei were able to hold the first ever video conference between two continents, encrypted using QKD. However, the satellite itself had access to the key, and could compromise the security of the conference if someone had access to it, but this problem was addressed in a paper published in June 2020, where improvements were made so that the satellite isn't required for any communication, it is only used to distribute the keys. This type of satellite is one of the first of it's kind, and will most likely be best used in combination with ground-based systems like quantum repeaters, which are trusted stations that decrypt and re-encrypt quantum keys for travel over long distances, as the errors from interactions between the photons and the fibers used to transport them increase dramatically with distance travelled. These technologies form the first steps in building a global quantum communications network.

Another interesting system is Kak's three-stage protocol, a quantum version of the well-known double lock system. Classically, the sender, Alice, encrypts her message using a private key, which the recipient, Bob, then also encrypts using his own private key. Bob sends the message back to Alice, she removes her encryption, then sends it back to Bob, and finally he removes his encryption and can read the message. The keys must be commutative, i.e. the order in which they are used doesn't matter. An attacker can see the message before and after it's encrypted, and can possibly use this information to reverse engineer the private keys. Subhash Kak proposed an entirely quantum version using photons polarized at random angles to encrypt the message, which would theoretically be resistant to attackers.

Wordcount: 990

References

1. *In 1950, Alan Turing Created a Chess Computer Program That Prefigured A.I.,* M. Stezano, history.com, 2018.
2. *History of Chess Computer Engines,* Mistreaver, chessentials.com, 2019.
3. *Brief History of Computer Chess,* TBS Staff, thebestschools.org, 2020.
4. *Programming a Computer for Playing Chess,* C. Shannon, Philosophical Magazine Ser.7, Vol. 41, No. 314, 1950.
5. *One Giant Step for a Chess-Playing Machine*, S. Strogatz, The New York Times, 2018.
6. *DeepMind's protein-folding AI has solved a 50-year-old grand challenge of biology*, W. D. Heaven, MIT Technology Review, 2020.
7. *The game has changed. AI Triumphs at solving protein structures*, R. F. Service, Science, 2020.
8. *Quantum computer*, W. C. Holton, Brittanica, 2005.
9. *Quantum Logic Gates*, NIST, 2018.
10. *Quantum Error Correction: An Introductory Guide*, J. Roffe, Contemporary Physics, 2019.
11. *Hands-On with Google's Quantum Computer,* N. Savage, Scientific American, 2019.
12. *Hello quantum world! Google publishes landmark quantum supremacy claim,* Nature, 2019.
13. *Google's Quantum Computer Achieves Chemistry Milestone,* N. Savage, Scientific American, 2020.
14. *Physicists in China challenge Google's 'quantum advantage',* P. Ball, Nature, 2020.
15. *Claude Elwood Shannon (1916-2001),* S. W. Golomb et al, American Mathematical Society, 2002.
16. *New Directions in Cryptography,* W. Diffie and M. E. Hellman, IEEE Transactions on Information Theory, 1976.
17. *How a quantum computer could break 2048-bit RSA encryption in 8 hours,* MIT Technology Review, 2019.
18. *Report on Post-Quantum Cryptography,* L. Chen et al, NIST, 2016.
19. *Communication Theory of Secrecy Systems,* C. Shannon, Bell System Tech Journal, 1949.
20. *China Reaches New Milestone in Space-Based Quantum Communications,* K. Kwon, Scientific American, 2020.
21. *Quantum Cryptography Demystified: How It Works In Plain Language,* D. Cardinal, extremetech.com, 2019.

All websites last visited on 07/12/2020.