The Evolution of Computing: AlphaGo

by Jim X. Chen
George Mason University

Google's AlphaGo program made headlines recently when it played a game of Go with a top human player and won. We're conditioned to like challenges until we face a formidable opponent, but what happens when we face a computer?

I learned to play Go when I was in elementary school. It's the simplest board game concept, yet it's the most difficult to play. I quickly gave up because it can take hours to finish just one game, and it takes seemingly forever to improve your skills. It's worth learning the game, but I wouldn't recommend it to all youngsters.

Figure 1 shows the Go board—on the 19 × 19 array, two players (black and white) take turns placing a stone of their own color on an empty grid point. Adjacent stones of the same color along the vertical and horizontal grid lines form an area. When an area is surrounded by an opponent's stones, it's "alive" if it has at least one empty point inside the area; otherwise, the stones in the area are captured and removed from the board. However, if the surrounded area has only one empty point left, it's captured because an opponent can place a stone in that point and solidly surround it. Therefore, an area is only really alive if it has two separate empty points inside—for example, as Figure 2 shows, if black

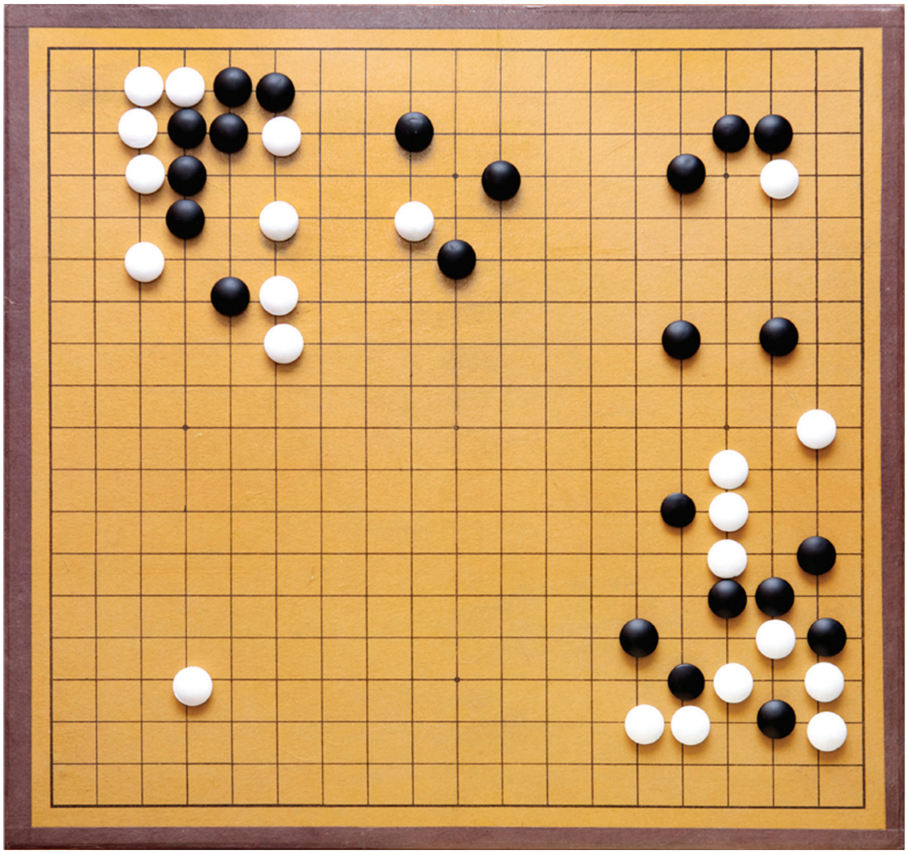

Figure 1. A Go board. On the 19 × 19 array, two players take turns placing a stone of their own color on an empty grid point.

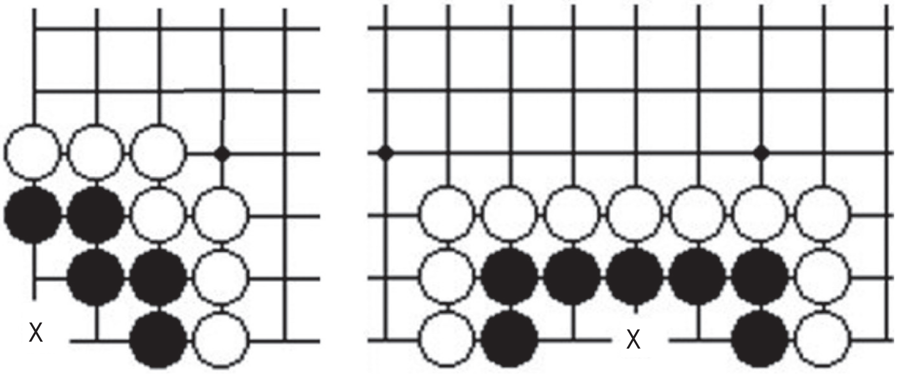

Figure 2. A critical point. If black places a piece at X, its area is alive. If white places a piece at X, black stones will eventually be removed.

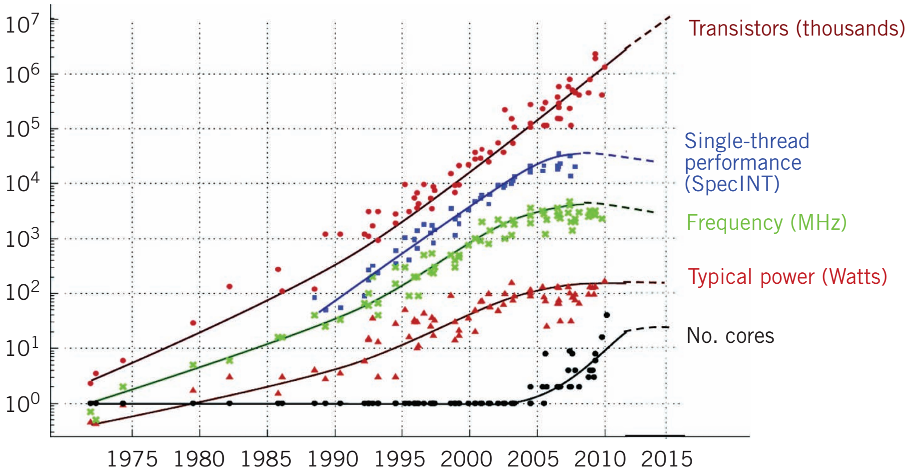

Figure 3. Microprocessor data over time. The number of transistors, integer performance, CPU frequency, power consumption, and number of cores has shifted—in some cases, dramatically. Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten Dotted line extrapolations by C. Moore.

places a piece at X on a border or corner, then its area is alive. If white places a piece at X, the black stones will eventually be removed. The players count points in their own live areas to decide who won or lost. That's it—simple! But rather frustratingly, it's difficult to see where to place a stone to gain a territorial advantage.

When I started my MS in computer science back in 1983, I looked at the game again in an artificial intelligence course. Roughly speaking, the person who starts a match has 19 × 19 possible empty points to place his or her stone, leaving 19 × 19 – 1 empty points for his or her opponent's next move. This goes on to searching (19 × 19)! = 361! possible combinations for the best results. Compared to other games, the Go board has many more options—for example, a chess board has only (8 × 8)! = 64! possible combinations. Furthermore, a chess piece's move is confined to its next step, as the king

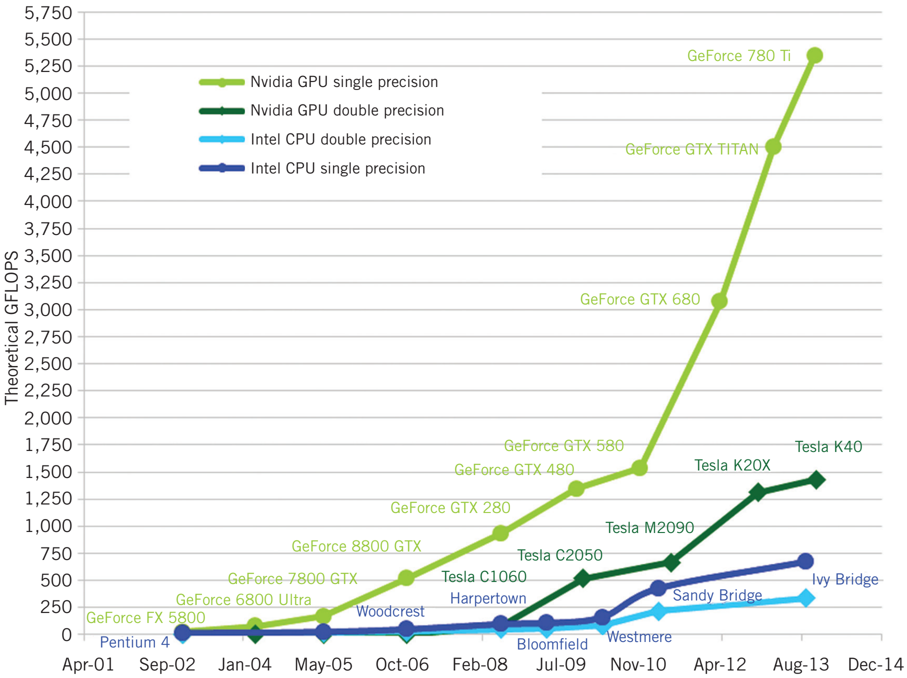

Figure 4. GFLOPS in GPU versus CPU over time. Here, GPUs are measured in GFLOPS instead of number of cores, which integrates speed instead of just the number of parallel processing units.

can only move one spot away from its current position, significantly reducing its number of combinations.

So where does this leave humans versus computers? The huge number of options in Go (361!) is still beyond computing power; ordinary human players can see just a few moves ahead. In fact, let's assume a human player can see three steps ahead—if so, we have at most 361 × 360 × 359 = 46,655,640 situations to consider, which is a manageable number for computers. Good human players can memorize successful gambits played in other games and maybe see a few moves further ahead, but machines can remember *all* existing games, although it takes a tremendous effort to pull that information together. AlphaGo was initially trained to mimic human play by attempting to match the moves of expert players from recorded games (its database holds 30 million moves). Once it reached a certain degree of proficiency, it was trained further by playing multiple games against other instances of itself, using reinforcement learning to improve its play. This accomplishment is all the more amazing because there's no easy way to evaluate a position in Go—the number of stones on the board is a weak indicator of a position's strength, and a territorial advantage is difficult to calculate. A professional human player can make relatively easier judgments compared to a machine due to instincts that algorithms can't capture, but AlphaGo might have broken this barrier.

But kudos to AlphaGo aside, the real story here is the evolution of computing. In 1983, CPU speed or frequency was 25 MHz. Assuming 1,000 CPU cycles correspond to making a decision on a Go board, it would have taken 31 minutes to make a decision about 361 × 360 × 359 = 46,655,640 possible moves back then. Today, a single CPU's frequency is above 4 GHz, as shown in Figure 3, making it 160 times faster on the same algorithm—in this case, taking less than 12 seconds to make that same decision.[1] In Figure 3, you can see that although CPU frequency stalls at its current level, the number of cores and transistors continues to double every other year or so. If we design

Erratum

In "Lesbian, Gay, Bisexual, Transgender, and Queer Students' Sense of Belonging in Computing: An Intersectional Approach," which appeared in *Computing in Science & Engineering,* vol. 18, no. 3, 2016, pp. 24; doi: 10.1109/MCSE.2016.45, the originally published version of the article contains an error in the author bio section. The correct text should read "Heather Wright is a research associate for the Computing Research Association's Center for Evaluating the Research Pipeline (CERP). Her research interests are centered on her passion for promoting social justice among all groups of individuals—especially those who are underrepresented in computer science and other closely related fields. Wright received a BS in sociology from Radford University, with minors in technical & business writing and women's studies. Contact her at heather@cra.org."

Computing in Science & Engineering regrets this error.

better computing algorithms, we could double the computing speed at its current level every other year or so.

In computing, a better measurement of computing power and performance is FLOPS (floating-point operations per second). Most microprocessors today can carry out 4 FLOPS per clock cycle, thus a single-core 2.5-GHz processor has a theoretical performance of 10 billion FLOPS, or 10 GFLOPS. Today, the most stunning recent hardware advancement is the GPU, which is akin to having thousands of tiny CPU cores for parallel processing. Figure 4 shows the evolution of the GPU compared to the CPU. Here, GPUs are measured in GFLOPS instead of number of cores, which integrates speed instead of just the number of parallel processing units (for example, the Nvidia GeForce GTX TITAN Z has 5,760 cores at 8,122 GFLOPS). If Go moves could be decided in parallel, we could end up with 0.01 second for making a move. Visualize that for a moment: from 31 minutes to 0.01 second in 30 some years.

AlphaGo is probably most powerful at its database, which is the weakest aspect of human capacity compared to computers, not in its evaluation. It'll be interesting to see what would happen if we removed the database. Regardless, with its advancement in computing and storage power, AlphaGo beats the most diligent and deeply intelligent human brains. The ramifications on the next evolutionary computing step remain to be seen. ■

Reference

1. W. Harrod, "A Journey to Exascale Computing," *Proc. High Performance Computing, Networking, Storage and Analysis,* 2012, pp. 1702–1730.

Jim X. Chen is a professor of computer science at George Mason University. He's also an associate editor in chief of *CiSE.* Chen's research is in graphics, virtual reality, and medical imaging; he has a PhD in computer science from the University of Central Florida. Contact him at jchen@gmu.edu.