DOUBLE IS BETTER
The Spracklens
on multiprocessor systems

In PLY 1/90 the Swedes report on a speed comparison in 24 test sessions between an Elite No.2 (= Fidelity Elite Avant Garde version 2) and the dual-processor version. The result: the Elite No.5 (= Fidelity Elite Avant Garde version 5) was once even 8% slower, then again up to 118% faster than the single-processor version! (The average speed increase was 59%.) Where do these different values come from? Dan and Kathe Spracklen answer this question themselves in the manual. We found their brief introduction to the problems of multiprocessor systems so interesting that we asked Fidelity for permission to reprint it, which was granted. We thank the company for their courtesy.

Dan & Kathe Spracklen: If you let the Elite No.5 calculate the same positions once in single processor mode and then in dual processor mode, you will find that the dual processor version is about 1.7 times as fast as the single processor. You may wonder why it is not exactly twice as fast, since two processors share the work. So let's make a few comments below about how the new Elite dual processor works.

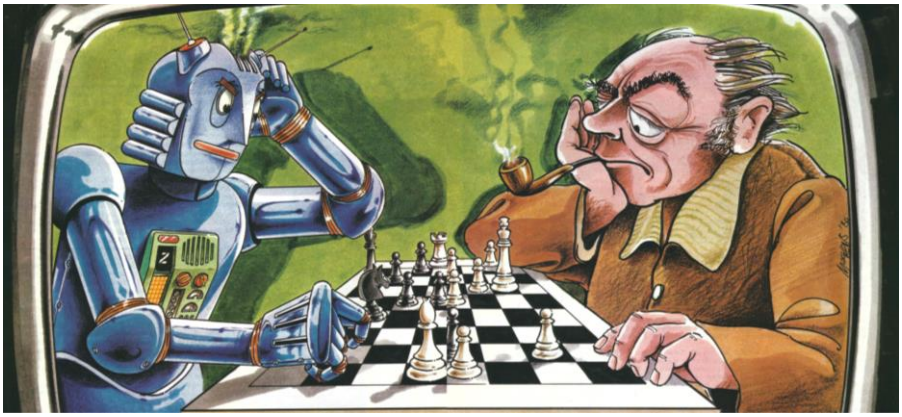
Let's start with a little terminology! The processor that registers the keystrokes and lights up the LEDs on a chess computer is also called the "master". The processor that assists him in selecting a move is called a "slave". The process of selecting a train is called "search". The task of the "master" is to distribute the search, i.e. to carry out the search work together with the "slave" and to decide on the selection of the best move. In most positions there are many possible moves, and the search must decide which is the best.

A simple and obvious way to divide the search would be to have the master and the slave each work on one move. As soon as one of them has finished his move, he could start on the next move on the list, and so on until all the moves have been examined. This method is quite applicable and gives a speed increase of about 1.4 compared to a processor working alone. Why so little?

The reason lies in the nature of the alpha-beta algorithm. Without going into long discussions of this procedure, one can say in a simplified way that the first move that is examined takes considerably more time than any of the other moves. The reason for this is that the examination of the first move yields a score that can be applied to the shortened processing of the following moves. If the move list is split, with master and slave each working on different moves, then both processors lack this comparative value and are condemned to a long and difficult job.

That means a lot of useless effort, and the end result is anything but impressive. It would also do no good to let the slave wait idly until the master has calculated through the first move; the result would be about the same.

A far more effective method of dividing the search has been proposed by Professor Tony Marsland of the University of Alberta. His method, which is also used by us, is based on doing the work of the tedious first move calculation together and only then dividing up the rest of the moves. In this method the master instructs the slave to work on certain parts of the search tree needed for the calculation of the first move. The result is the remarkable speed-up to 1.7 times achieved by our dual processor.



"Overhead" is the loss of time or performance that occurs when using a particular programming method. In a multiprocessor system, there are four types of "overhead", all of which contribute to the inability to achieve the ideal speed increase of a factor of 2.

1.) COMMUNICATION: This is the time lost because the two processors have to contact each other and exchange information. One processor must create messages to inform the other of the direction and results of the search work. The other processor must receive and respond to these messages. This is called the **"COMMUNICATION OVERHEAD"**.

2.) In SYNCHRONISATION: To understand this concept, we must consider what happens when the search is almost over. Suppose that the master is just thinking about the last move and the slave about the penultimate move. One of the two will finish earlier than the other and will be "unemployed" afterwards. The time that one processor has to spend idly waiting for the other is called the **"SYNCHRONIZATION OVERHEAD"**.

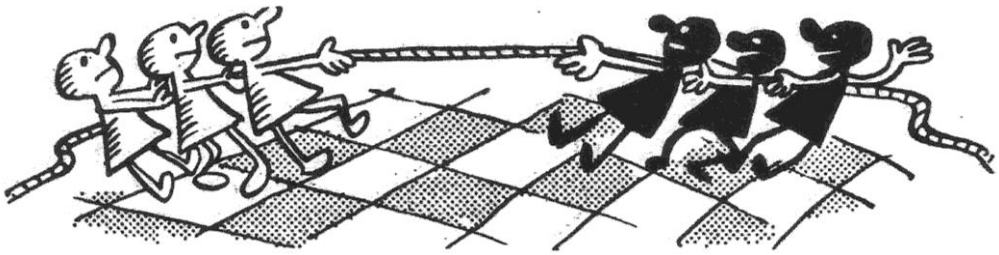
3.) In DIVISION OF WORK: Let's think again about what we said about dividing the search work by the "you take one, I'll take one" method. We mentioned that the reason for the relatively low efficiency of this method is that the score found in the examination of the first move speeds up the processing of the following moves. The better this first score is, the faster the other moves will be processed. If, for example, our first move wins the opponent's queen, we do not need to dwell long on the other moves, which either win less material or none at all. If, on the other hand, our first move only leads to a piece being positioned a little better, then we have to deal with the other possible moves in more detail.

Fidelity Elite Avant Garde version 5

Two 68000 processors, 16 MHz, 192K RAM »hash tables«. The world's first commercially available multiprocessor. The increased number of »hash tables« serves to optimise the cooperation between the two processors.

(photo copyright © by http://www.schaakcomputers.nl/) (600 dpi)

As a result of the constant sorting of the train list, in many cases the first train is also the best. Sometimes, however, a move further down the list turns out to be better. When this happens, the evaluation found in the first move is less useful for processing the further moves than the evaluation of the new move. The processor that found the better move will therefore immediately continue with the new evaluation. The other processor, on the other hand, must continue with the old value until it has finished processing the current move. The performance loss caused by the fact that the new evaluation is not immediately available to both processors is called **"DIVIDED SEARCH OVERHEAD"**.



An interesting side effect of this overhead is seen in tests based on solving mating problems. The basic idea in using mating problems to measure the performance of chess computers is that there is an obvious, but hard to find, winning move. The tester can measure the time it takes the computer to find the winning move, and this usually gives a fairly reasonable measure of how fast and efficient the chess computer is. However, chess problems by their very nature consist of positions in which the best move almost never coincides with the first move examined. They are therefore positions with artificially increased division of labour overhead! For this reason, programmers who want to test the performance of their split search should better use arbitrary positions from master games than work with mate problems where the multiple processor does not show its best side.

4.) When splitting HASH TABLES: a hash table is a large area of memory in which a processor can store information gained during the search process. If the same position comes up again in another variant, the processor can look up the result in the hash table instead of having to start the calculation all over again. Hash tables are particularly effective in endgames, where the search is many times faster with their help than without them. The reason for this is that positional repetitions occur particularly often in the endgame.

The **"SPLIT HASH OVERHEAD"** is caused by the fact that each processor has its own hash table. The master cannot look into the slave's table and the slave cannot look into the master's table. Therefore, if one of them reaches a position that is already stored in the other's hash table, he still has to start the calculation all over again. In the opening, the middlegame and even the early endgame, this makes no decisive difference. But when the number of pieces on the board is greatly reduced, the hash table overhead can drastically reduce the effectiveness of a multiple processor system. In some cases, dual-processor systems for pawn endgames even take longer than single-processors!

It is precisely this crippling effect of the SPLIT TABLE OVERHEAD that has led us to search for a way out. While keeping the sales department at bay with one arm, we looked for and found a solution: if splitting the hash tables was causing the problem, we thought, why not work with COMMON hash tables? In this design, only one table would be used (and that would be the master's, because it is bigger). The disadvantage of this would be increased communication overhead because the two processors would have to exchange more information about the hash tables.

In the hardware configuration originally intended for the dual processor, this method would not have been applicable, because neither processor had the ability to "tap the other on the shoulder" and say "I need information about the hash tables" or "I have information about the hash tables for you". Each processor would have wasted so much time waiting for the other to acknowledge his request that the resulting communication overhead would have been worse than that caused by the shared hash tables. However, the method could work if the slave had a way to interrupt the master at any time by "interrupting" to retrieve or store hash tables information.

While we were fighting with the sales department for deadline extensions, we sat down with the technical department. Yes, an ingeniously simple design modification would allow us to use interrupts! As they say: the rest is history! Instead of slowing down in the endgame, the dual processor can now proudly point to a speed-up by a factor of 1.5 to 1.6.

Now you have an idea of how hard the software and hardware experts at Fidelity have worked to make computer chess even more enjoyable. We hope you will enjoy the exciting advances in multiprocessors as much as we have enjoyed bringing them to you.

Authors: Dan & Kathe Spracklen

Source: Modul (German magazine) No. 2, June 1990: Doppelt hält besser – Die Spracklens über Mehrprozessor-systeme. Translated from the original German article into English.
Photo copyright © by http://www.schaakcomputers.nl/

For the original German article (text), see: https://www.schaakcomputers.nl/hein_veldhuis/database/files/06-1990,%20Modul,%20Doppelt%20halt%20besser%20-%20Die%20Spracklens%20uber%20Mehrprozessor-systeme.pdf

First Published on November 19, 2021 - Size: 14.142 KB