Gustav 4, 128 GB RAM - výsledky testování 288 samomatů 10.tahem
Gustav 4, 128 GB RAM - results of testing of 288 selfmates in 10 moves

(Václav Kotěšovec, 6.12.2019)

V databázi WinChloe bylo v červnu 2019 celkem 288 ortodoxních samomatů 10. tahem, které nebyly testovány počítačem ("C?"). Rozhodl jsem se je všechny postupně testovat. Tomuto projektu jsem věnoval půl roku počítačového času. Použil jsem program Gustav 4.0 g (jeho 64-bitovou verzi), jehož autorem je Olaf Jenkner. Úlohy jsem testoval na mém počítači se 128 GB paměti RAM pod operačním systémem Windows 7 64-bit.

Processor:	Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz
Installed memory (RAM):	128 GB
System type:	64-bit Operating System

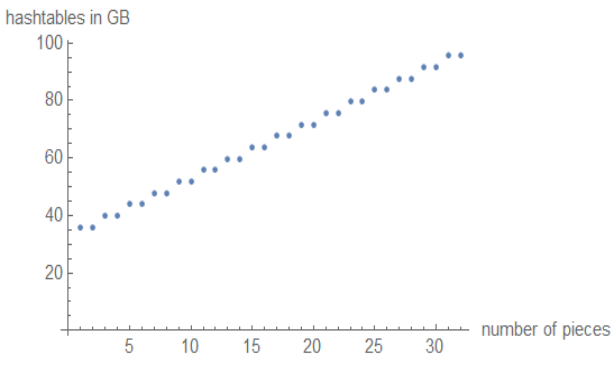
In the WinChloe database were in June 2019 a total of 288 orthodox selfmates in 10 moves, which were not computer tested ("C?"). I decided to test them all gradually. I devoted half a year of computer time to this project. I used program Gustav 4.0 g (its 64-bit version) by Olaf Jenkner. I tested the selfmates on my computer with 128 GB of RAM under Windows 7 64-bit.

Dal jsem si časový limit cca 100 hodin pro každou úlohu (odhadovaný čas jsem zjistil metodou popsanou dále), řada úloh se však řešila v desítkách minut nebo jednotkách hodin. **Celkový čas testování byl přes 3800 hodin**. Z celkového počtu 288 úloh se ukázalo jako netestovatelných z časových důvodů pouze 8 skladeb ! To bylo pro mě příjemným překvapením, očekával jsem, že takových úloh bude mnohem víc. Z toho 5 úloh má odhad 200-300 hodin každá, to by ještě bylo možné testovat, ale 3 úlohy jsou daleko nad touto mezí. Pro počítač je časově nejnáročnější tato úloha s odhadovaným časem přes 2000 hodin.

I took a time limit of about 100 hours for each selfmate (I estimated the time using the method described below), but many selfmates were solved in tens of minutes or units of hours. **The total testing time was over 3800 hours**. Out of 288 selfmates, only 8 s#10 proved to be untestable for time reasons! It was a pleasant surprise for me, I expected that there would be many more selfmates. Of these, 5 selfmates have an estimate of 200-300 hours each, it would still be possible to test, but 3 selfmates are far above this limit. For a computer, this s#10 is time consuming, with an estimated time of over 2000 hours.

Program využíval až cca 90 GB paměti RAM. Jelikož můj počítač má 8 jader, paralelně zde mohly běžet ve zbytku paměti jiné (obvykle nešachové) programy, ale program Gustav byl spuštěn vždy jen jeden, ale s parametrem "[Hashtables] max", kdy využíval maximální možnou paměť pro HashTables.

The program used up to about 90 GB of RAM. Since my computer has 8 cores, other (usually non-chess) programs could run in parallel in the rest of the memory, but program Gustav was only running one, but with the "[Hashtables] max" parameter, using the maximum possible memory for HashTables.



V programu Gustav 4.0 (64-bit) je maximální možná paměť závislá na celkovém počtu kamenů (jak ukazuje graf vlevo). To je rozdíl třeba od programu Popeye, který by uměl využít paměť celou, na tento typ úloh je však výrazně pomalejší než Gustav.

In Gustav 4.0 (64-bit), the maximum possible memory is dependent on the total number of pieces (as shown in the graph left). This is different from the Popeye program, which is able to use all memory, but for this type of problems is much slower than Gustav.

Korektní / nekorektní
C+ / cooks

Celkem **38 úloh bylo nekorektních** a další 3 měly chybu v pozici. Je třeba zdůraznit, že výsledky jsou metodou Brute Force ("[Parameter] bf"), tj. hrubou silou a úlohy, které obstály v tomto testu je tak možno označovat jako "C+". Daleko více si cením právě těchto korektních úloh. Důkaz korektnosti vyžaduje úplný test a ten je obvykle velmi časově náročný, zatímco řada vedlejších řešení (pokud existují) se najde rychleji.

A total of **38 selfmates were cooked** and the other 3 had an error in position. It should be pointed out that the results are Brute Force ("[Parameter] bf") and the selfmates that have passed this test can be referred to as "C+". I appreciate these correct selfmates much more. Proof of correctness requires a complete test, and it is usually very time consuming, while a number of cooks (if any) are found more quickly.

Podařilo se nalézt vedlejší řešení i v jedné úloze z Alba FIDE.
New cook in the FIDE Album: F185 FIDE Album 2001-2003

Program Gustav umožňuje v extrémně rychlém čase i částečné testy (v režimu "[Parameter] auto", případně při manuálním zadání různých parametrů). Nejprve jsem proto projel všech 288 úloh částečným testem v režimu "Auto". Už v této fázi bylo nalezeno 16 nekorektních úloh (a 3 chybné pozice), tedy 50% nekorektností bylo objeveno již v této fázi. V případě 224 úloh byl částečný test v režimu "Auto" kratší než 1 minuta! Pouze u 27 úloh byl částečný test delší než 1 hodina.

The Gustav program also allows partial tests in extremely fast time (in "[Parameter] auto" mode, or when entering various parameters manually). First of all, I passed all 288 selfmates with a partial test in "Auto" mode. Already in this phase, 16 incorrect selfmates (and 3 wrong positions) were found, i.e. 50% of cooks were discovered at this stage. For 224 selfmates, the partial test in "Auto" mode was less than 1 minute! Only for 27 selfmates the partial test was longer than 1 hour.

Zde je jeden nesmírně zajímavý výsledek (faktor, který jsem před lety testoval i v případě Alybadixu, kde jsou možné také částečné testy při parametru Y1). Pokud úloha projde částečným testem (tato fáze je rychlá, ale nelze takovou skladbu označit jako "C+"), pak je zajímavé, kolik z takových skladeb se ukáže být přesto nekorektní ? Zde to bylo celkem 19 úloh (z toho 14 úloh mělo vedlejší řešení, 5 úloh mělo duály). Odtud vychází, že **pravděpodobnost, že úloha částečně testovaná programem Gustav, je opravdu korektní, je** asi 100 * 19 / (288 - 3 - 16 - 8) = **92.7 %**

Here is one extremely interesting result (a factor that I tested years ago in the case of Alybadix, where partial tests with parameter Y1 are also possible). If the selfmate passes a partial test (this phase is fast, but it cannot be marked as "C+"), then it is interesting how many of these selfmates prove to be cooked yet? Here it was a total of 19 selfmates (of which 14 selfmates had an unintended solution, 5 selfmates had duals). Hence, the **probability that the selfmate partially tested by Gustav is really correct is** about 100 * 19 / (288 - 3 - 16 - 8) = **92.7%**

K tomu je možno ještě doplnit, že citovaná nekorektní úloha z Alba FIDE patří právě do skupiny úloh, kde částečný test projde a nekorektnost neobjeví. Nutný je test "Brute Force".

In addition to this, the cook from Album FIDE belongs to a group of problems where a partial test passes and the incorrectness does not appear. A "Brute Force" test is required.

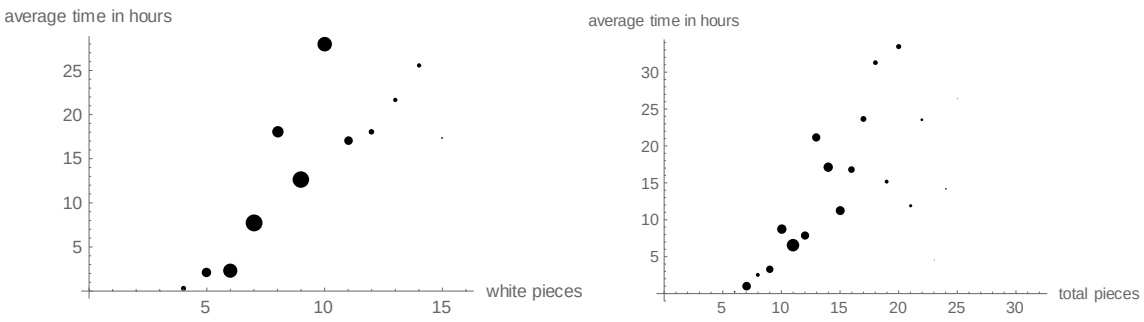
WinChloe / PDB

Výsledky jsem průběžně zaznamenával zpět do databáze WinChloe (a jednou měsíčně posílal Christianu Poissonovi do jeho centrální databáze) a současně jsem je zadával i do databáze PDB (pokud tam příslušná úloha byla). Z této činnosti vznikla zajímavá statistika. Příjemně mě překvapilo, že z celkového počtu testovaných úloh chybělo v databázi PDB pouze 34 skladeb, tj. 88% jich bylo i v této databázi (Frank Müller zadal hodně úloh do PDB databáze). Z tohoto počtu bylo celkem 117 s#10 v PDB již označeno jako "C+" nebo "cooked" (je třeba ovšem poznamenat, že šlo převážně o časově méně náročné úlohy). Výsledky mého testování byly v případě celkem 126 skladeb v databázi PDB nové. Můžete si je prohlédnout zde

I continuously recorded the results back to the WinChloe database (and sent Christian Poisson to his central database once a month) and at the same time entered them into the PDB database (if the selfmate was there). Interesting statistics have emerged from this activity. I was pleasantly surprised that out of the total number of tested problems, only 34 s#10 were missing in the PDB database, i.e. 88% were in this database too (Frank Müller added a huge amount of selfmates). Of these, a total of 117 s#10 in the PDB have already been identified as "C+" or "cooked" (but it should be noted that these were predominantly less time consuming selfmates). The results of my testing were new for 126 selfmates in the PDB database. You can see them here: problems which I recently tested.

Na doplnění několik grafů
Several charts

Závislost času řešení na počtu bílých kamenů / na celkovém počtu kamenů.
Dependence of the solving time on the number of white pieces / on the total number of pieces.



Velikost bodů v grafu ukazuje váhu toho kterého výsledku (velké body odpovídají většímu počtu testovaných úloh se shodným počtem kamenů). Jak se dalo očekávat, čas potřebný k přezkoušení narůstá s počtem bílých kamenů. Silnější černá protihra (spec. černá dáma) naopak čas testování zkracuje. Průměrné časy řešení jsou pro s#8 8.6 minuty, s#9 81 minut a s#10 14.3 hodiny. Tyto hodnoty však mají **velký rozptyl**, takže je třeba je brát jen jako orientační.

The size of the points in the graph shows the weight of each result (large points correspond to a larger number of tested selfmates with the same number of pieces). As expected, the time needed for testing increases with the number of white pieces. On the other hand, a stronger black counterpart (special a black queen) shortens the testing time. Average solving times are for s#8 8.6 minutes, s#9 81 minutes and s#10 14.3 hours. However, these values have a large variance, so they should be taken for guidance only.

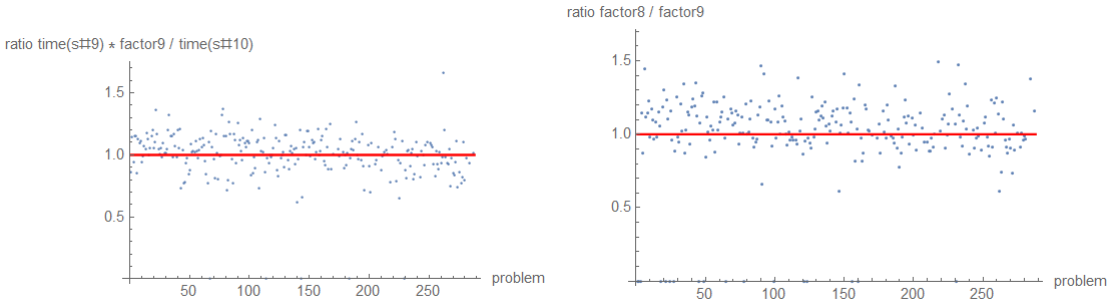
Program Gustav během řešení ještě ukazuje spotřebovaný čas při nižším počtu tahů a podle toho sám počítá faktor (poměr času mezi úrovní n+1 a n), ze kterého se dá udělat poměrně **dobrý odhad celkového času řešení**. Dále budu označovat factor8 = čas(s#8) / čas(s#7), factor9 = čas(s#9) / čas(s#8). Odhadovaný čas pro s#10 je potom factor9 * čas(s#9), případně factor8^2 * čas(s#8). Příklad z programu Gustav (64-bit version):

During the solving, Gustav shows the time consumed at a lower number of moves and accordingly calculates a factor (the ratio of time between level n+1 and n) which can be used to make a relatively **good estimate** of the total solving time. I will refer to factor8 = time (s#8) / time (s#7), factor9 = time (s#9) / time (s#8). The estimated time for s#10 is then factor9 * time (s#9) or factor8^2 * time (s#8). Example from Gustav (64-bit version):

```
Kein Matt in   5:  00 : 00 : 00 : 03
Kein Matt in   6:  00 : 00 : 00 : 39  Faktor 11.89
Kein Matt in   7:  00 : 00 : 07 : 46  Faktor 11.92
Kein Matt in   8:  00 : 00 : 57 : 53  Faktor  7.45
Kein Matt in   9:  00 : 06 : 44 : 02  Faktor  6.98
```

Je zajímavé, že obecně se ukazují tyto odhady mírně nadhodnocené, tj. finální čas je většinou o trochu menší než odhadovaný.

Interestingly, these estimates generally show slightly overestimated, i.e. the final time is usually slightly smaller than the estimated one.



V 57.2% případů je čas odvozený z času pro s#9 krát faktor9 větší než skutečný čas pro s#10.
V 59.4% případů je faktor8 > faktor9.

In 57.2% of cases, the time derived from the time for s#9 * factor9 is greater than the real time for s#10.
In 59.4% of cases, factor 8 > factor 9.

Vysvětlit se to asi dá použitou velkou pamětí, kde při dlouhém testování program zde častěji nachází pozice, kterými se již během testování jednou zabýval a to více častěji čím déle úlohu řeší. Tento efekt pravděpodobně nebude platit, pokud se použije menší paměť.

It can be explained by the large memory used, where during long testing, the program more often finds the positions it has already dealt with during testing, and more often the longer the problem solves. This effect may not apply if less memory is used.