

Stelvio

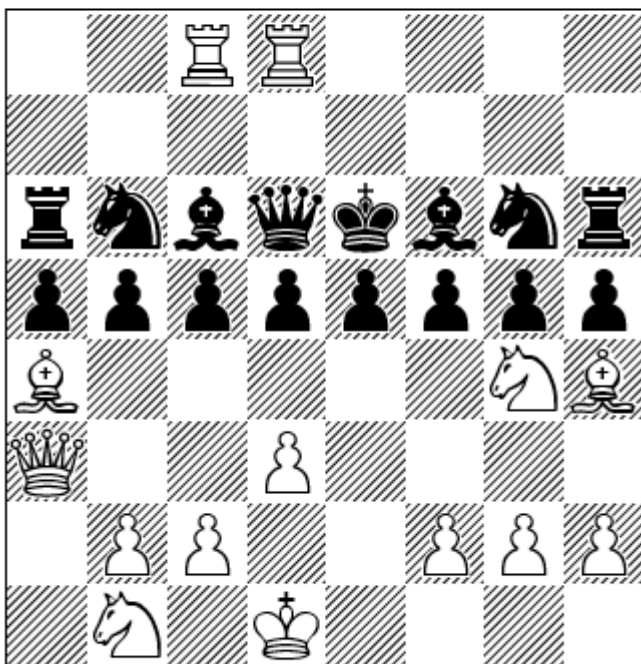
Introduction

Stelvio is a program devoted to solving orthodox SPGs (Shortest Proofgames). To make things bit clearer, its easiest to consider an example. Let us look at a masterpiece from Finland.

Unto Heinonen

Probleemblad 1998

1. Prize



SPG in 23.0 moves

This can be checked by Stelvio, and in less than 1 minute (depending on hardware and other parameters). Stelvio comes to the conclusion, that there is exactly one way to reach this position in 23.0 moves:



The successful strategy and the unique solution can subsequently be displayed:



The name

Stelvio, with its 2757m of altitude above sea level, is one of the highest and one of the most beautiful mountain passes of the Alps. As a bike rider, I've been up there many times and have good memories of it. I needed a name, and I liked the way Stelvio sounds. This name breaks the

"tradition" of naming the program after a famous mathematician like Jacobi or Euclide. Let's say the idea of naming it Scholze was not very appealing...



Compatibility

As Stelvio is written in Java, it can run on various platforms, among them Windows, Mac and Linux.

Requirements

Solving SPGs is often memory-intensive, especially for SPGs with a lot of free moves. For that reason, it can be very beneficial to run Stelvio with most of the RAM that you have on your machine. You can find the amount of RAM you have in the System properties, common values for a notebook would be 8g-32g. I'd say 4g is bare minimum (can Tetris run with less?).

Btw: Running Stelvio with way more RAM than is needed for a particular SPG will likely increase the solving time for this SPG, as there are costs involved accessing large amounts of memory on hardware level. You can witness this e.g. in the required ramp-up time: When given a lot of memory, Stelvio needs much longer to get going. This is due to the fact that a large cache needs to be initialized. But for SPGs that take a long time to solve, and added 20 seconds or so up front do not matter. When giving Stelvio almost all or all of your RAM, it is advised to close other applications so that the memory is in fact free to use.

Installation

As Stelvio is written in Java, there needs to be a corresponding Java Runtime Environment installed for it to run. You can get your OS-specific version from here: <https://www.oracle.com/ch-de/java/technologies/downloads/>. I have tested Stelvio with Java 11 and Java 17. E.g. for Windows, you can

download Java 17 with this link: https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi. You then need to run the downloaded file and install it to a directory of your choice, recommended is "c:\java\java17" (this path should not contain spaces). Now download stelvio<version>.zip and unzip it to a directory of your choice, e.g. "c:\spg\stelvio". The zip file contains the following:

- bin folder containing some *.jar files (the code)
- doc folder containing PDFs (documentation)
- stelvio<version>.bat (the file to run)
- stelvio<version>.sh (the file to run on Linux)
- stelvioUI.ini (the parameter file)
- problems.txt (example input)

Adjust stelvio<version>.bat

You should open stelvio<version>.bat in a text editor and adjust the maximum memory settings. Per default, it says "-Xmx8g" in the file, which means Stelvio is allowed to use 8g of RAM. If you have more memory available, then give most of it to Stelvio, e.g. "-Xmx16g" for 16g of RAM. Also make sure that the path that points to the Java Runtime Environment is correct. The default is "c:\java\java17\", but you need to adjust this in case you opted to install Java at some other location.

To test your installation, just double-click stelvio<version>.bat, which should run Stelvio with the provided problems.txt file.

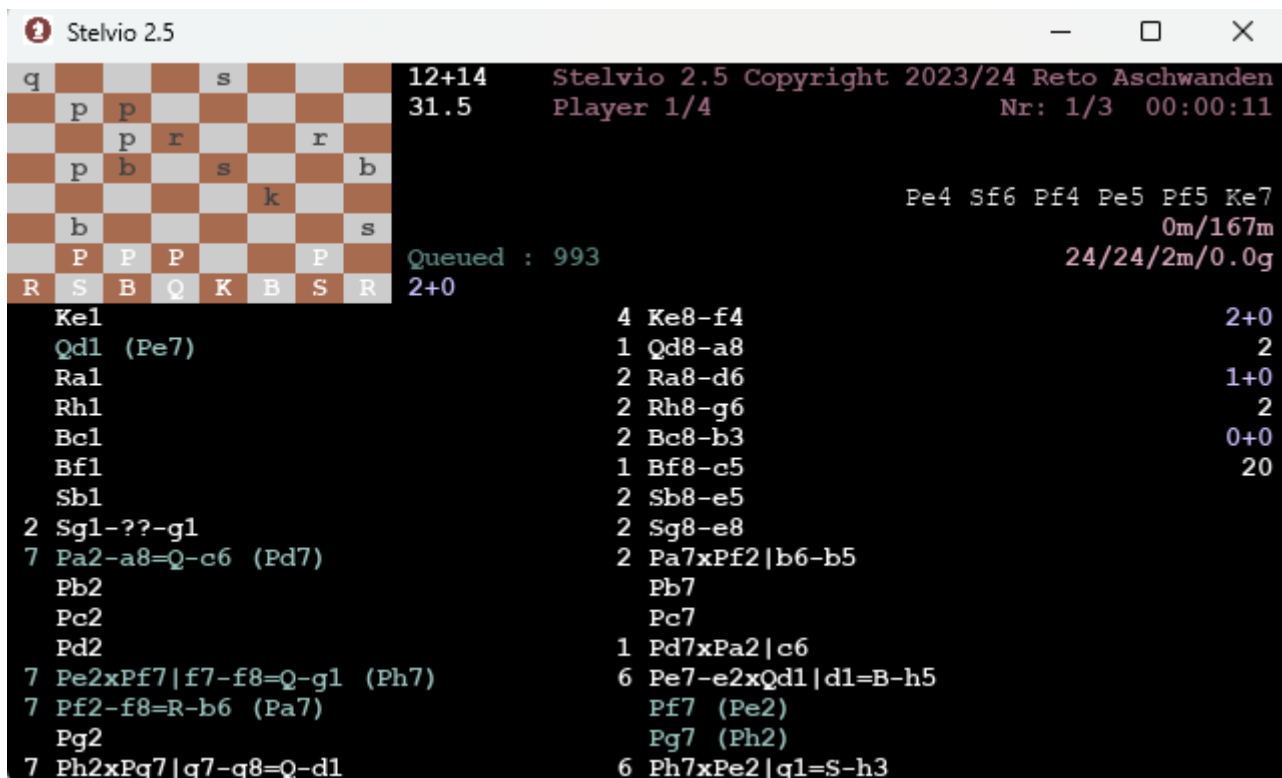
Troubleshooting

If Stelvio does not start when double-clicking on the stelvio<version>.bat file, then most likely the Java Runtime Environment cannot be found or its version is too old. Check if the path contained in stelvio<version>.bat pointing to java.exe is correct and adjust if needed.

UI

The simple terminal-based UI gives some impression where Stelvio stands in the solving process.

Here is what is displayed for a strategy player:



What it all means:

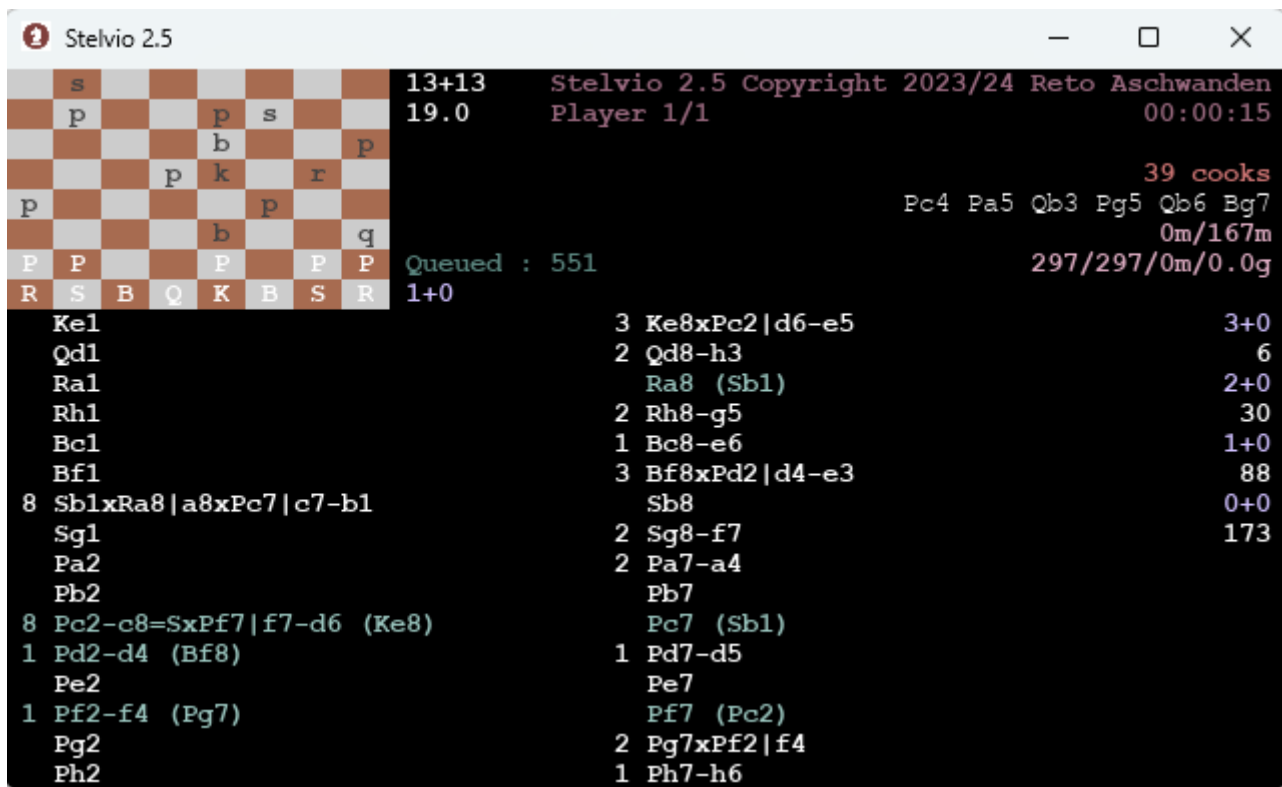
- Below the copyright notice:
 - Player 1/4: Which page is being displayed. We are currently looking at the page for strategy player 1 out of 4 such players. Pages can be switched using left-arrow/right-arrow in order to see pages for other strategy players/strategy seekers.
 - Nr: 1/3: The SPG being solved. In the example, there are 3 SPGs contained in the input file, and we are currently solving the first SPG.
- Next to the board:
 - Number of pieces and number of moves are obvious.
 - Queued: 993: Number of strategies that have been found that still need to be played. The numbers here are before any possible strategy splitting.
 - 2+0: Number of free moves for the current strategy by color, so 2 for white and 0 for black.
- On the right:
 - Solving time so far (hh:mm:ss).
 - Current move path (not to be taken too seriously, this can be stale/inconsistent data for technical reasons. If garbage is displayed, this has no effect on the solving itself, only the display is invalid).
 - Cache/strategy metrics:
 - 0m/167m: Currently, the playing cache is basically empty (0 million). Cache can hold 167 million positions (this is actually just an approximation, but good enough). This cache is bigger if you have more RAM available, which can be very useful depending on the SPG in question.
 - 24/24/2m/0.0g:

- 24: The number of strategies already taken off the strategy queue (before any strategy splitting).
 - 24: The strategy number being played (possibly after strategy splitting). In the current case, no strategy splitting was needed, and therefore the numbers coincide.
 - 2m: Number of moves played for the current strategy in millions.
 - 0.0g: Total number of moves played for all strategies so far, in billions. Currently, less than 100m have been played, so this displays 0.
- Below the board:
 - The current strategy, i.e. what each piece does, white on the left and black on the right, with associated move count per piece.
 - The column on the far right: Histogram of already played strategies (top 8 entries, as there is limited space). So in the example, 2 strategies with 2+0 free moves have already been played, next to strategies with 1+0 and 0+0 free moves.

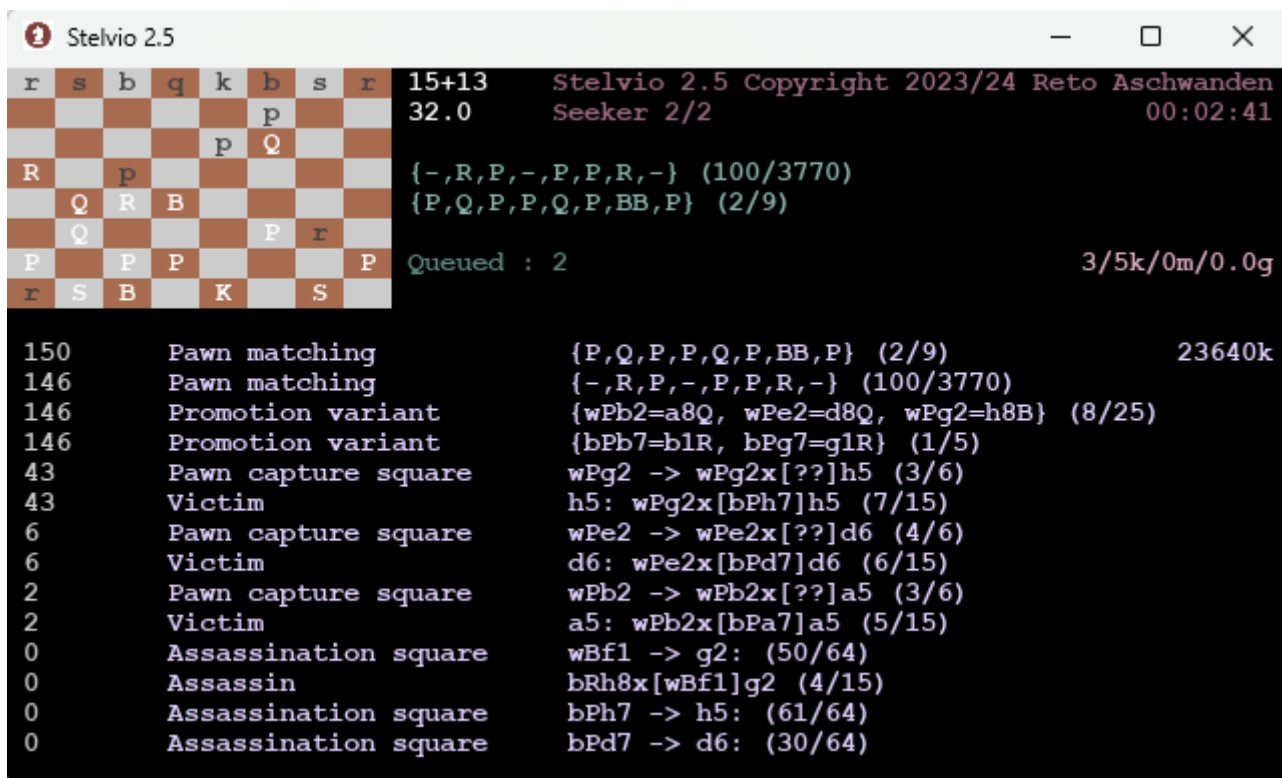
As already displayed above, when a solution is found, pressing up-arrow/down-arrow displays the solution and the successful strategy.

When there are cooks, Stelvio counts the number of strategies that contribute to cooks, in the below example 39 cook strategies were found so far. In case strategies which can be uniquely played are found, these are counted as valid solutions and displayed as well.

A brief note on what Stelvio counts as a cook/solution: In case a strategy can be played in multiple ways, then this counts as a cook strategy for Stelvio. This is the correct approach to take for 99.9% of the SPGs out there. In case a SPG requires an even number of half-moves, e.g. 22.0, then any solution/cook needs to have an even number of half-moves. Correspondingly for SPGs requiring an odd number of half-moves. This seems a good approach to take, since who is at play is part of the position to reach. So in case in the example one could reach the diagram position in 21.5 moves, then this does not count for Stelvio. Any shorter move-path with same parity of half-moves does count though, so in the example, Stelvio would count solutions/cooks in {21.0, 20.0, 19.0, ... 1.0} moves.



Above we have seen the UI for a strategy player, let's now look at what is displayed for a strategy seeker:



- Next to the board:
 - The green lines: How the pawns are assigned to partitions currently in the strategy seeking process. Partitions are the types of pieces on the board that are mutually exclusive, we have partition values (K, Q, R, WB, BB, S, P). WB stands for white square bishop and BB for black square bishop. Top line is black assignment {-,R,P,-,P,P,R,-}, lower line is white assignment {P,Q,P,P,Q,P,BB,P}. This means for white, that a/c/d/f/h pawns remain pawns, b/e-pawns are

queens in the diagram position and g-pawn is a black square bishop in the diagram. For black, c/e/f pawns remain pawns, the b/g-pawns are rooks visible in the diagram and a/d/h pawns are captured. At the outset of strategy seeking, all possible assignments of pawns to partitions are calculated, in the example we have 9 for white and 3770 for black. During strategy seeking, these assignments are all gone through in a big loop, and we are currently at assignment 2 for white and 100 for black. This information gives you a hint how far strategy seeking and therefore solving already is.

- Queued : 2. The number of strategies in the queue that are waiting to be played. This queue is the link between all the strategy seekers and all the strategy players.
- On the right:
 - Solving time so far (hh:mm:ss).
 - Strategy metrics:
 - 3/5k/0m/0.0g:
 - 3: The number of strategies that were retained after strategy analysis and which therefore need to be played.
 - 5k: The number of strategies initially found (before strategy analysis), in thousands.
 - 0m: Since the strategy seeker is not playing, this number remains 0.
 - 0.0g: Since the strategy seeker is not playing, this number remains 0.

From version 1.6 onwards, the current strategy seeking path can be displayed. This is helpful in order to get an estimate of how long the seeking process will take. The first 14 levels of seeking are shown, together with how long the corresponding seek node has been active (in seconds, the number on the left). This information is somewhat analogous to the first 6 moves displayed when playing, only here, we have the first 14 "seeking moves". The information should be read top-down, so the two pawn matching choices have been active for 150 seconds respective 146 seconds. The next thing the seeking algorithm looks for are promotion variants. After that, a pawn capture square is chosen for the Pg2, currently we chose h5. Btw, the amount of possible options displayed to the right is sometimes only an estimate, but better an estimate than nothing.

If you are in histogram mode, then up-arrow/down-arrow switches between histogram and this new seeking information. Gathering this information takes a bit of time and in case you are not interested, it can be fully turned off. See the StelvioParameters file for details.

Histogram mode

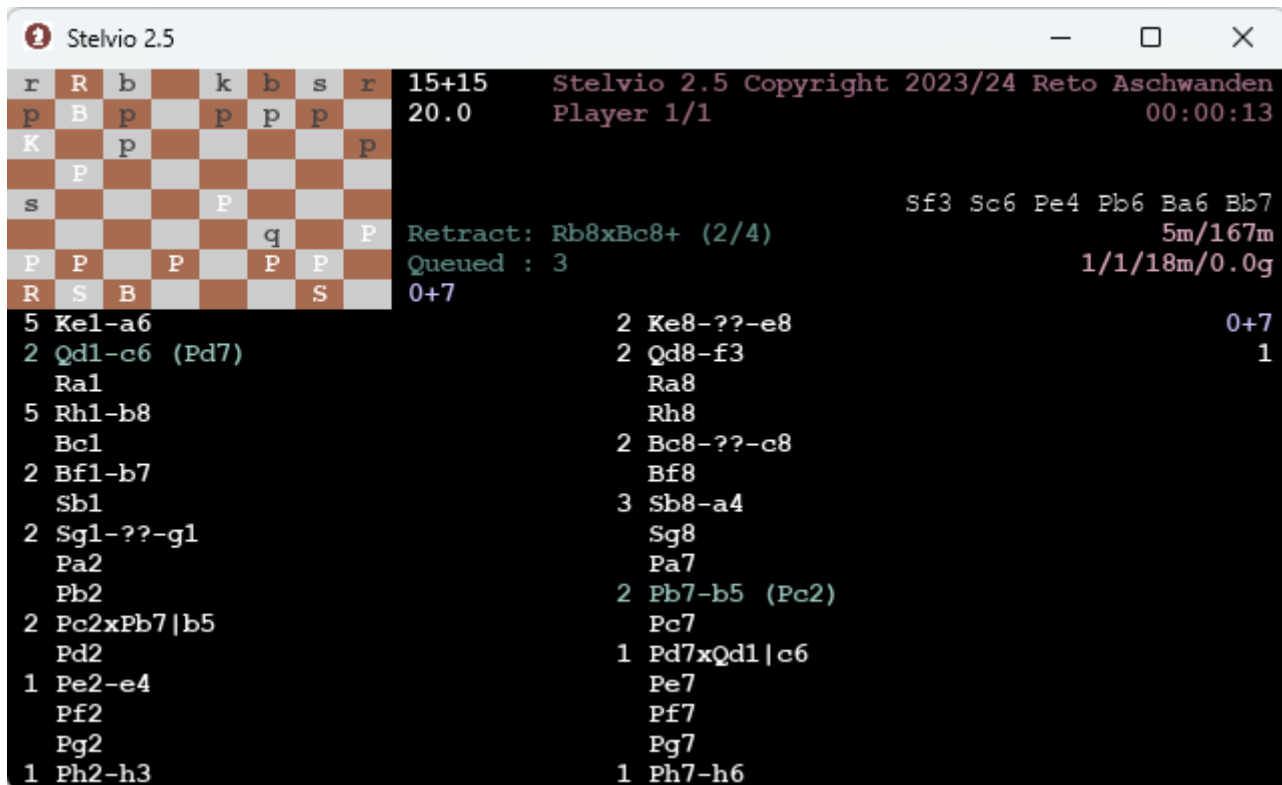
In order to get a first impression if an SPG can be solved in a reasonable amount of time, it can be useful to simply search for all the strategies without actually playing them. This is what the histogram mode is for. In this mode, strategies are added up and grouped by white/black free moves.



So in the example, there are 72 strategies currently found by this strategy seeker (numbers are per seeker) with 2+0 free moves, 138 with 1+0 and 403 with 0+0 free moves. When all seekers are finished, the strategy numbers are added up and written into the output file. It is also possible to attain a partial histogram: You can tell Stelvio to only start adding up strategies after strategy number X. Beware though that strategy numbers are only deterministic in case only 1 strategy seeker is used. This can be useful if you want to know what is left in terms of strategies, if Stelvio already solved up to strategy X.

Last move retraction

In case a king is in check in the diagram, it can be very helpful to retract the last half move and solve all implied shorter SPGs and combine their results thereafter. By default, this is what Stelvio now does in such a case. While solving, the retracted move is displayed next to the diagram, as well as how many such moves there are. In the example, there are four possible last moves, and we have currently retracted the second. The problem displayed is the one with the move retracted. In the special e.p. retracted case, 2 half moves are retracted, as these 2 half moves cannot be split up. In the special castling retracted case, Stelvio makes sure that the king and involved rook are not moving beforehand. In histogram mode, the histogram of all sub-SPGs are written to the output file, as well as an overall histogram. Beware that this overall histogram will not coincide with the histogram you get when solving the SPG without retraction. This retraction functionality can be switched off by a new parameter named `retractionMode`. Limitation: Last move retraction and save/read strategies from/to file are not currently supported in combination. Btw: The `retractionMode=always` (i.e. even if king is not in check) will probably be added in a future release.



Some thoughts on parallel solving

Both histogram and normal solving are executed in parallel. Modern computers come with several CPUs, so calculating in parallel can be beneficial for performance.

The basic architecture is quite simple: There are x strategy seekers and y strategy players. The numbers x and y can be parametrized to suit the SPG at hand and your hardware. When running in parallel, the strategy seekers search for strategies and put the found strategies into a queue. The strategy players on the other hand consume strategies from this queue and try to play them. The granularity for the strategy seekers is a single pawn partition matching combination plus some further few seeking choices. This can be a bit coarse, and I might adjust this in the future. In the example below, there are $9 * 3770$ such combinations, which the seekers consume one by one until none are left.

```

Stelvio 2.5
15+13  Stelvio 2.5 Copyright 2023/24 Reto Aschwenden
32.0   Seeker 2/2 00:02:41

{ -,R,P,-,P,P,R,- } (100/3770)
{ P,Q,P,P,Q,P,BB,P } (2/9)

Queued : 2 3/5k/0m/0.0g

150 Pawn matching {P,Q,P,P,Q,P,BB,P} (2/9) 23640k
146 Pawn matching { -,R,P,-,P,P,R,- } (100/3770)
146 Promotion variant {wPb2=a8Q, wPe2=d8Q, wPg2=h8B} (8/25)
146 Promotion variant {bPb7=b1R, bPg7=g1R} (1/5)
43 Pawn capture square wPg2 -> wPg2x[??]h5 (3/6)
43 Victim h5: wPg2x[bPh7]h5 (7/15)
6 Pawn capture square wPe2 -> wPe2x[??]d6 (4/6)
6 Victim d6: wPe2x[bPd7]d6 (6/15)
2 Pawn capture square wPb2 -> wPb2x[??]a5 (3/6)
2 Victim a5: wPb2x[bPa7]a5 (5/15)
0 Assassination square wBf1 -> g2: (50/64)
0 Assassin bRh8x[wBf1]g2 (4/15)
0 Assassination square bPh7 -> h5: (61/64)
0 Assassination square bPd7 -> d6: (30/64)

```

Parallel solving examples

The screenshots below are from version 1.27, but the insights gained are still valid in the current version.

Same SPG but different concurrency metrics

I give a few examples of the effect of parallelizing, when it is beneficial and when it is not. Let's take the 2023 Andernach TT winner by Michel. In non-parallel mode (the only mode available in Stelvio 1.2), this SPG is solved on my notebook in some 35 minutes:

```

Stelvio 1.27
12+13  Stelvio 1.27 Copyright 2023 Reto Aschwenden
16.5   00:35:39

{ S,P,P,P,P,Q,P,P } (14/14)
{ P,R,-,P,P,P,P,P } (3/3)

Done. Found 1 solution
478184/33932k/0m/4.4g

2 Ke1-??-e1 4 Ke8-f5 6+0
  Qd1 (Rh8)  Qd8 4
4 Ra1-g7 Ra8 5+1
  Rh1 7 Rh8xPf2|f2xBf1|f1xQd1|d1xSg1|f3 1
  Bc1 Bc8 (Pa2) 5+0
  Bf1 (Rh8) 2 Bf8-e3 53
  Sb1 Sb8 4+1
1 Sg1-f3 (Rh8) Sg8 28
8 Pa2xPb7|b7xBc8|c8=QxPc7|c5-d1 Pa7 3+2
  Pb2 Pb7 (Pa2) 40
  Pc2 1 Pc7-c5 (Pa2) 2+3
  Pd2 Pd7 21
  Pe2 Pe7 1+4
  Pf2 (Rh8) Pf7 5
  Pg2 1 Pg7-g6 4+0
  Ph2 1 Ph7-h5 730

```

Both strategy seeking and strategy playing take significant time for this SPG, so splitting up is beneficial. With 1 seeker and 1 player, the solving time drops to 27 minutes in Stelvio 1.3:

```
Stelvio 1.27
r s q s
p p p R
p
k p
b r
P P P P P P
S B Q K R
12+13 Stelvio 1.27 Copyright 2023 Reto Aschwenden
16.5 Player 1/1 00:27:18
{S,P,P,P,P,Q,P,P} (14/14)
{P,R,-,P,P,P,P,P} (3/3)
{Queued: 0}
Done. Found 1 solution
478184/478k/0m/4.4g
2 Ke1-??-e1 4 Ke8-f5 6+0
Qd1 (Rh8) Qd8 4
4 Ra1-g7 Ra8 5+1
Rh1 7 Rh8xPf2|f2xBf1|f1xQd1|d1xSg1|f3 1
Bc1 Bc8 (Pa2) 5+0
Bf1 (Rh8) 2 Bf8-e3 53
Sb1 Sb8 4+1
Sg1-f3 (Rh8) Sg8 28
8 Pa2xPb7|b7xBc8|c8=QxPc7|c5-d1 Pa7 3+2
Pb2 Pb7 (Pa2) 40
Pc2 1 Pc7-c5 (Pa2) 2+3
Pd2 Pd7 21
Pe2 Pe7 1+4
Pf2 (Rh8) Pf7 5
Pg2 1 Pg7-g6 4+0
Ph2 1 Ph7-h5 730
```

As no strategy requires a lot of memory, we can also split up strategy playing without harm. With 1 seeker and 2 players, the solving time drops to 20 minutes:

```
Stelvio 1.27
12+13      Stelvio 1.27 Copyright 2023 Reto Aschwenden
16.5       Player 2/2                                00:20:01

{S,P,P,P,P,Q,P,P} (14/14)
{P,R,-,P,P,P,P,P} (3/3)
{Queued: 0}

Done. Found 1 solution
232959/232k/0m/2.2g

0+0

2 Ke1-??-e1      4 Ke8-f5      6+0
  Qd1             Qd8          3
2 Ra1-f4 (Rh8)   Ra8          5+1
  Rh1             5 Rh8xBf1|f1xRa1|f4xSg1|f3 1
  Bc1             Bc8 (Pa2)    5+0
  Bf1 (Rh8)       2 Bf8-e3      25
  Sb1             Sb8          4+1
1 Sg1-f3 (Rh8)   2 Sg8xPf2|f6-g8      13
9 Pa2xPb7|b7xBc8|c8=RxPc7|c6-g7  Pa7          3+2
  Pb2             Pb7 (Pa2)    21
  Pc2             1 Pc7-c6 (Pa2)  2+3
  Pd2             Pd7          12
  Pe2             Pe7          1+4
3 Pf2-f6 (Sg8)   Pf7          3
  Pg2             1 Pg7-g6      4+0
  Ph2             1 Ph7-h5      362
```

In the above configuration, the queue is sometimes empty. That means that the players have nothing to do at times, so an additional seeker will likely reduce the solving time. With 2 seekers and 2 players, the solving time drops to 18 minutes:

Stelvio 1.27

12+13 Stelvio 1.27 Copyright 2023 Reto Aschwarden
16.5 Player 2/2 00:18:19

{S,P,P,P,P,Q,P,P} (14/14)
{P,R,-,P,P,P,P,P} (3/3)
{Queued: 0}

Done. Found 1 solution
240371/240k/0m/2.2g

0+1

2 Ke1-??-e1	4 Ke8-f5	6+0
Qd1	Qd8	4
4 Ra1-g7	Ra8	5+1
Rh1	5 Rh8xBf1 f1xPf2 f6xSg1 f3	1
Bc1	Bc8 (Pa2)	5+0
Bf1 (Rh8)	2 Bf8-e3	25
Sb1	2 Sb8xPa2 b8	4+1
1 Sg1-f3 (Rh8)	Sg8	16
7 Pa2xPb7 b7xBc8 c8=QxPc7 c7-b8 (Sb8	Pa7	3+2
Pb2	Pb7 (Pa2)	18
Pc2	Pc7 (Pa2)	2+3
Pd2	Pd7	10
Pe2	Pe7	1+4
3 Pf2-f6 (Rh8)	Pf7	3
Pg2	1 Pg7-g6	4+0
Ph2	1 Ph7-h5	367

Just for fun, I tried with 4 seekers and 4 players, which still shaves a minute of solving time:

Stelvio 1.27

12+13 Stelvio 1.27 Copyright 2023 Reto Aschwarden
16.5 Player 1/4 00:16:56

{S,P,P,P,P,Q,P,P} (14/14)
{P,R,-,P,P,P,P,P} (3/3)
{Queued: 0}

Done. Found 1 solution
118282/118k/0m/1.1g

1+0

2 Ke1-??-e1	4 Ke8-f5	6+0
Qd1	2 Qd8xPa2 c7-d8	1
4 Ra1-g7	Ra8	5+1
Rh1	6 Rh8xSg1 f3xBf1 f1xPf2 f6-f3	1
Bc1	Bc8 (Pa2)	5+0
Bf1 (Rh8)	2 Bf8-e3	15
Sb1	Sb8	4+1
1 Sg1-f3 (Rh8)	Sg8	9
6 Pa2xPb7 b7xBc8 c8=RxPc7 c7 (Qd8)	Pa7	3+2
Pb2	Pb7 (Pa2)	8
Pc2	Pc7 (Pa2)	2+3
Pd2	Pd7	7
Pe2	Pe7	1+4
3 Pf2-f6 (Rh8)	Pf7	1
Pg2	1 Pg7-g6	4+0
Ph2	1 Ph7-h5	174

As my notebook now runs out of hardware, increasing the numbers further will have a negative effect, since in the end, all seekers/players compete for the same underlying hardware.

Different SPGs and the usefulness of parallelism for them

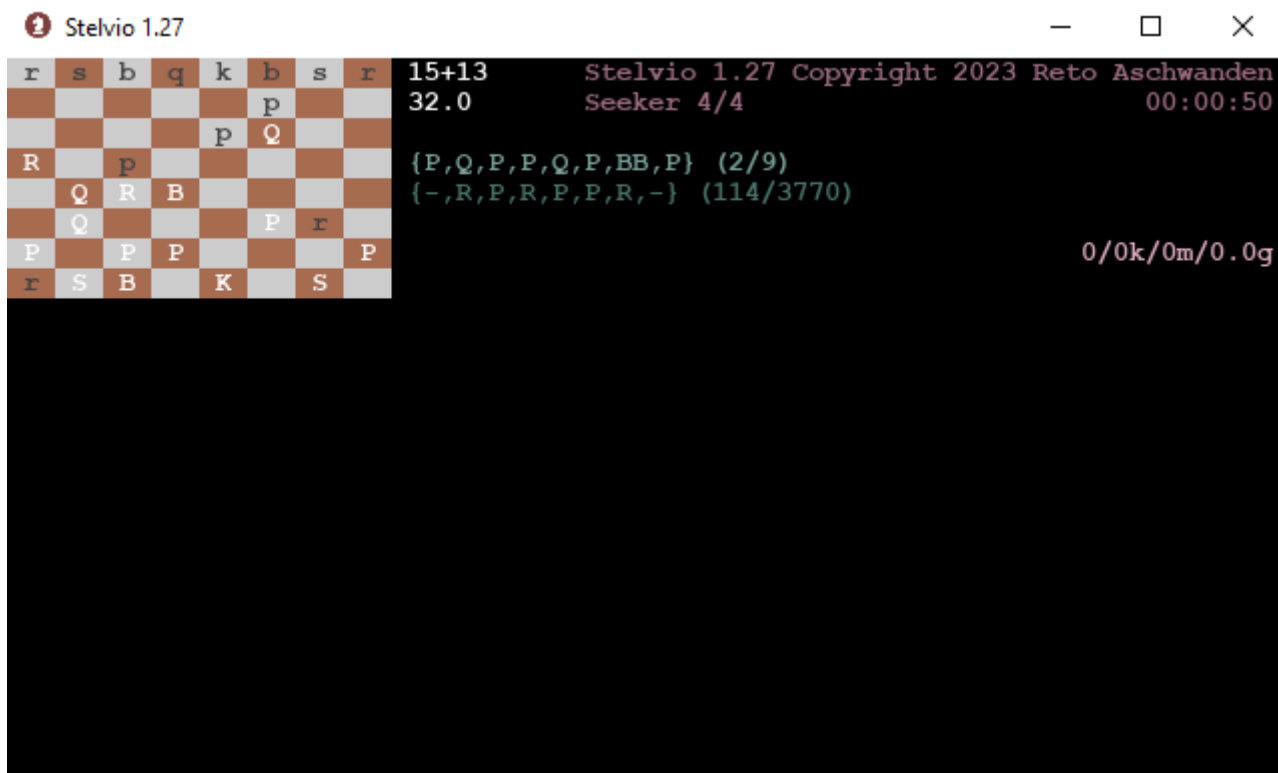
Histogram mode in parallel

The seekers consume all pawn partition matching combinations one by one. In case there are only

few of these, then parallelism is not very useful. E.g. in the following length record case (P1407171), there is only exactly one such pawn partition matching combination, so parallelism is useless, as only one seeker has work to do:

The screenshot shows the Stelvio 1.27 application window. On the left is a chessboard with pieces: s, B, q, S on the top row; K, p, p, p, p, Q on the second row; B, R, p on the third row; p, R, p on the fourth row; P, b, P, S, P on the fifth row; r, k, P, P, P on the sixth row; and s on the seventh row. To the right of the board, the text reads: '16+16 Stelvio 1.27 Copyright 2023 Reto Aschwanden', '41.5 Seeker 1/1', and '00:00:00'. Below this, two sets of pawns are listed: '{P,P,P,P,P,P,P,P} (1/1)' and '{P,P,P,P,P,P,P,P} (1/1)'. Further right, it says 'Done. Histogram completed' and '24/0k/0m/0.0g'. At the bottom left, a histogram is shown with three bars: '2+6 : 8', '2+5 : 8', and '2+2 : 8'.

Much better is the following case (P1000601) with $9 * 3770 = 33930$ pawn partition matching combinations. What usually happens though is that most of the strategy seeking time is required for one or very few such combinations. That has the negative effect that parallel seeking time does not proportionally decrease with the number of strategy seekers. In the worst case, 33929 pawn partition matching combinations could require basically no time and 1 such combination could require 99% of the time. Then parallelizing would not be very useful.



Solving in parallel

- Parallel solving can be detrimental under some circumstances, in case there are multiple players. The reason is that the players each need their own position cache, and this cache requires a lot of memory. So in case you have 2 players instead of 1, then each position cache is only half the size. Now in case while playing a strategy, the position cache is close to full, then solving performance for this strategy decreases dramatically. Therefore, it can be slower to play strategies in parallel than to play them serially. A case where this happens is the aforementioned length record problem (P1407171), which by the way still cannot be solved.
- In the Andernach TT winner above, parallel solving is pretty useful, cutting the solving time in half.
- Parallel solving is the most useful in case strategy seeking takes up most of the time and there are a lot of pawn partition combinations to comb through. As there is almost no limit for number of seekers (given enough available CPUs), you can go all-in in that respect. An order of magnitude can be shaved of the solving time in that way for P1386153.

Current limitations, possible future developments

Two things come to mind:

- Parallel playing of a single strategy.
- Parallelize not only on one machine, but across a cluster of machines.

Collision detection modes

With v2.0, Stelvio has two modes of collision detection: A more involved mode (`expensiveCollisionDetectionMode = on`) and a less involved mode (`expensiveCollisionDetectionMode = off`). Both modes have their merit, as both can be orders of

magnitude faster than the other, depending on the problem. In case `expensiveCollisionDetectionMode = default`, then the expensive mode is chosen for an SPG if and only if it has at least 28 pieces, which seems like a reasonable heuristic.

Collision detection is done for every found strategy. In case there are only few strategies to analyze, then the additional costs of more involved collision analysis will be negligible. One example where the involved analysis is very beneficial is the 4th Prize in the Champagne Tourney 2023 by Peter van den Heuvel.

The screenshot shows the Stelvio 2.0 chess engine interface. On the left is a chessboard with pieces: White (P, P, B, S, P, P, R, K) and Black (s, b, Q, p, p, p, b, k). The main display shows search statistics: 16+15, 21.0, and a list of moves: {P,P,P,P,P,P,P,P} (1/1) and {P,P,P,P,P,P,-,P} (1/1). It also indicates 'Done. Found 1 solution' and '4/0k/0m/0.0g'. The move list on the right includes: 4 Ke1-??-e1, 3 Qd1-d8, 2 Ra1, 2 Rh1-h6-b6, 2 Bc1-f4-d6, 2 Bf1-d3-c2, 1 Sb1-d2, 2 Sg1-f4, 1 Pa2, 1 Pb2, 1 Pc2-c3, 1 Pd2-d4, 1 Pe2-e3, 1 Pf2-f3, 1 Pg2, 1 Ph2xPg7|g3, 2 Kg8-g8-h7, 3 Qd8-b1, 2 Ra8-a5-f5, 1 Rf8-f8-h8, Bc8, 1 Bf8-g7, Sb8, 4 Sg8-d1, 2 Pa7-a4, 1 Pb7-b5, 1 Pc7-c5, Pd7, Pe7, Pf7, 3 Pg7-g3 (Ph2), and 1 Ph7-h6.

The involved analysis detects that the black queen must pass through e1, eliminating any free moves that white would otherwise have. In case this is solved with `expensiveCollisionDetectionMode = off`, then solving takes hours if not days.

Any massacre style SPG will be solved quicker with `expensiveCollisionDetectionMode = off`, as there are billions of strategies to analyze and involved analysis is not needed. But not only massacres can benefit from less collision detection, the following SPG (P1013138) by me, Michel and Gerd is also solved quicker with less analysis (by a factor 4).

```

Stelvio 2.0
13+11 Stelvio 2.0 Copyright 2023/24 Reto Aschwandten
18.5 Player 1/1 00:09:21

{P,P,P,P,P,-,-,P} (10/10)
{S,P,P,S,P,Q,P,-} (133/133)

Done. Found 1 solution
7152/77k/0m/0.0g
Queued : 0
0+1

2 Kg1-g1-h1 Ke8 0+6
2 Qd1-h3 Qd8 254
1 Ra1-g1 Ra8 0+5
4 Rf1-f1xPd7|c1xPh7|h5-h7 Rh8 1k
1 Bc1-e3 2 Bc8-b5 (Pc2) 0+4
Bf1 (Pf7) Bf8 7k
2 Sb1xPf7|f1 Sb8 0+3
1 Sg1-f3 Sg8 15k
Pa2 1 Pa7-a6 (Pc2) 0+2
Pb2 Pb7 24k
4 Pc2xBc8|b5xPa7|a6-a7 Pc7 0+1
1 Pd2-d4 7 Pd7xPe2|e2-e1=S-c1 (Rh1) 18k
Pe2 (Pd7) Pe7 0+0
1 Pf2-f4 6 Pf7xPg2|g2-g1=RxBf1|f1 (Sb1) 8k
Pg2 (Pf7) Pg7
Ph2 1 Ph7-h5 (Rh1)

```

Check protection

With v2.0, Stelvio can sometimes determine that check protection is needed. This can greatly reduce solving times. An example is the classic 1999 Phénix first prize by Michel (P1000010), which can now be solved in reasonable time on an ordinary machine.

```

Stelvio 2.5
15+13 Stelvio 2.5 Copyright 2023/24 Reto Aschwandten
25.5 Player 1/1 00:00:54

Pa4 Sc6 Ra3 Se5 Rc3 Pb5
54m/167m
3/7/243m/0.2g
Queued : 27
7+0

Ke1 3 Kc8-c8-a8 7+0
2 Qd1-f6 (Pg7) 2 Qd8-e3 (Pd2) 1
6 Ra1-b7-a1 3 Rd8-d8-f4 (Pg2) 6+0
Rh1 2 Rh8-g8-g2 1
Bc1 3 Bc8-h5 2+0
3 Bf1-a6 3 Bf8-h2 2
Sb1 3 Sb8-g4 1+0
Sg1 2 Sg8-??-g8 3
2 Pa2xPb7|b5 1 Pa7-a5
Pb2 1 Pb7-b5 (Pa2)
Pc2 1 Pc7-c6
2 Pd2xQd8|e3-e4 Pd7
Pe2 Pe7
Pf2 Pf7
3 Pg2xRa8|f4-f5 1 Pg7xQd1|f6
1 Ph2-h4 Ph7

```

By looking at the displayed strategy, it becomes apparent that check protection necessity for the black king was detected, as a white rook is planned to make a detour (Ra1-b7-a1). Check protection gives rise to strategy splitting: Once check protection necessity is detected, all the possible ways to

provide it are calculated. The original strategy is subsequently split into several sub-strategies. This splitting is visible in the UI in the metrics on the right hand side: In the example, we already took 3 strategies from the queue, but we are currently playing the 7th (sub-)strategy.

Btw: Strategy splitting can also arise from baseline collisions when it is unclear who makes a detour, e.g. Ra1-g1, Rh1-b1 is split into 2 sub-strategies.

User interaction

The playing of the current strategy can be stopped by typing 's'. Stelvio will move on to the next strategy thereafter. Pressing Ctrl-C cancels the solving process. In parallel mode, you have different UI pages per seeker/player. You can switch between these pages using left-arrow/right-arrow.

Input / Output

A simple text file (by default problems.txt) serves as input. It needs to be in the same directory as stelvio<version>.jar. In problems.txt, the SPG needs to be given in FEN notation on the first line, and the number of half moves on the second line, something like:

```
1nbq4/ppk1p3/Rp5p/3npr2/R3P3/2br1B1P/PP2P2P/1NBQNK2
65
```

Pieces are denoted by:

- K/k = King
- Q/q = Queen
- R/r = rook
- B/b = bishop
- N/n/S/s = knight
- P/p = Pawn

You can add several problems to the input file, which will then be solved in succession.

The result is written into an output file (by default named problems_out.txt). There is a stelvioUI.ini file for parameters that can be adjusted by the user. If no such file is present, then default values are used. See also StelvioParameters.pdf.

Read/write strategies to file

Stelvio can be advised to save all found strategies in structured format to disk. In a second step, instead of searching for strategies, Stelvio can thereafter read these strategies from disk and try to play them. This is especially useful in case one uses histogram mode at first. Calculating the histogram sometimes requires a lot of time, only to find very few strategies. In order to subsequently play these strategies, they can now be read from disk in no time, instead of recalculating all of them all over again. Limitation: Last move retraction and save/read strategies from/to file are not currently supported in combination.