

Feedback, suggestions

Stelvio can get a lot better in the future, and this is where you come in (if you want). By supplying information like the following:

- In the given SPG X, Stelvio gets stuck on this obviously impossible strategy Y.
- In the given SPG X, Stelvio does not see the collision between pieces Y and Z.
- In the given strategy X, Stelvio does not see that the rook needs 5 moves and only calculates 4.
- In the given SPG X, Stelvio plays Y strategies, but Natch/Euclide only play far less. That implies that Stelvio is missing some strategy analysis logic.
- etc
- Any other improvement ideas not already listed under future developments.

Any such information can be sent to [stelvio.feedback\(at\)gmail.com](mailto:stelvio.feedback(at)gmail.com). Do not hesitate to send any comments and improvement suggestions to this address. The more input I have, the better I can make an informed decision what to add for upcoming releases.

But beware that these two things will **not** be added in future releases:

- I will not waste my time creating a fancy UI. Not a single line of JavaScript will ever be added.
- No fairy elements will ever be supported. I programmed Stelvio with orthodox rules in mind, broadening the rules would be equivalent to rewriting most of the code.

But give me some slack: As I have probably invested north of 1000 hours for Stelvio already, I might need a bit of a break before more improvements will be added.

There will be bugs

Any software with this level of complexity contains bugs, Stelvio is certainly no exception. As a colleague of mine once nicely put it: If there are so many possibilities to make mistakes, you are bound to make some. To counteract this, I put a lot of effort into the testing machinery to try to detect mistakes. If you get an internal failure nonetheless, please check the `problems_out.txt` file first to see if it is not a simple out-of-memory issue. Those can be usually fixed by giving Stelvio more RAM. If it is not a memory issue (and not an invalid input on your side), please send a bug report to [stelvio.feedback\(at\)gmail.com](mailto:stelvio.feedback(at)gmail.com) with the `problems_out.txt` file and a screenshot attached.

The good news is that despite there being bugs, the likelihood that Stelvio comes up with a wrong C+ verdict is very low. For that to happen, Stelvio would have to find the intended solution but miss all the cooks. Usually, a cooked problem is cooked for several reasons, so Stelvio would have to miss all of these, which is highly unlikely.

Known Issues

There are some limitations concerning what Stelvio can do. Most internal structures do not grow in size greater than to a certain extent, but a few do. In case a problem exceeds assumed boundaries, there will be some internal error, possibly `OutOfMemoryError/ArrayIndexOutOfBoundsException`

or similar. One case that this can happen is when there are many rooks of the same color on the board. Say in a case like the 57.5 move length record problem by Frolkin/Pronkin. The reason is, that at the beginning, all possible permutations are calculated that determine which rook is which piece. Not only which pawn promoted to a rook matters, but also the promotion square. For internal reasons, a rook that castled is different to a rook that did not, which further adds to the permutation explosion. In the 57.5 case, this permutation structure gets huge and does not fit into the RAM of an ordinary PC. To make a long story short, I made a few assumptions in terms of size of the different components, usually very generous assumptions. If an SPG nonetheless breeches those size barriers, then you will most likely get some sort of internal error. You can submit the internal error as a bug report, and I'll check if I need to rethink the sizing of certain components.

Known limitations

Stelvio is not well suited for massacre SPGs. In case there are a lot of captures with relatively few moves, then a simple brute force approach (like Jacobi or Popeye use, as far as I know) can be more efficient. There are at least two reasons for this:

- There are an enormous amount of strategies to find, making Stelvio's approach very costly.
- When playing moves in brute force manner, it is quite simply to know when to stop playing: You can stop when you reach a point where one side needs to play more captures than there are moves left.

I do not have any intention of adding massacre specific logic to Stelvio, as I think massacre SPGs are usually not artistic or interesting enough (with a few notable exceptions, like the double rex solus SPG).

I have further found that Stelvio is not fast for SPGs with mandatory cross captures. This necessity is detected rather late in the solving process, which means Stelvio plows through gazillions of strategies without cross captures, only to find out that for all of them, there are not enough moves due to some line being permanently blocked. I have made some algorithmic inroads on this, but all of this is still beta and cannot be used yet.

Future developments (hopefully)

- Improve strategy analysis.
 - When a king is in check in the diagram position, then the consequences of this should be used.
 - Add check protection logic.
 - Finish the started cycle detection logic.
 - etc.
- Improve strategy playing
 - Especially the logic around mutually obstructing trajectories.
- Parallelize Stelvio
 - Splitting up strategy searching

- Play strategies from a pub/sub queue, either within one machine or even across a cluster of machines.
- Improve caching logic. Empirical data needs to be gathered here first.
- User defined conditions for strategies (partial solving)
- Other improvements triggered by user feedback.

Many thanks go to

- Pascal Wassong and Etienne Dupuis for their excellent programs. Especially Pascals achievement with Natch cannot be underestimated, since there was no predecessor to look up to.
- Gerd Wilts for the PDB export.
- Daniel Bühler and Silvan Diem at my company for granting me the disused hardware that now grinds away in my basement.
- Silvio Baier and Thierry LeGleuher for testing.
- Thomas Brand for allowing me to host Stelvio on his website.
- Special thanks to Michel Caillaud for very extensive testing and loads of valuable feedback.