Søren Riis *is a Computer Scientist at Queen Mary University of London. He has a PhD in Maths from University of Oxford. He used to play competitive chess (Elo 2300).*

A Gross Miscarriage of Justice in Computer Chess

by Dr. Søren Riis

Introduction

In June 2011 it was widely reported in the global media that the International Computer Games Association (ICGA) had found chess programmer International Master Vasik Rajlich in breach of the ICGA's annual World Computer Chess Championship (WCCC) tournament rule related to program originality. In the ICGA's accompanying report it was asserted that Rajlich's chess program Rybka contained "plagiarized" code from Fruit, a program authored by Fabien Letouzey of France.

Some of the headlines reporting the charges and ruling in the media were "Computer Chess Champion Caught Injecting Performance-Enhancing Code", "Computer Chess Reels from Biggest Sporting Scandal Since Ben Johnson" and "Czech Mate, Mr. Cheat", accompanied by a photo of Rajlich and his wife at their wedding.

In response, Rajlich claimed complete innocence and made it clear that he found the ICGA's investigatory process and conclusions to be biased and unprofessional, and the charges baseless and unworthy. He refused to be drawn into a protracted dispute with his accusers or mount a comprehensive defense. In his only public statement on the matter to date he said

> *Rybka does not include game-playing code written by others, aside from standard exceptions which wouldn't count as 'game-playing'.*

Rajlich added that the ICGA's action would not deter him from continuing to improve Rybka and sell it commercially, as he has done successfully for the past five years, and that his program would be willing to compete in a match or tournament with any worthy challenger.

This article re-examines the case. With the support of an extensive technical report by Ed Schröder, author of chess program Rebel (World Computer Chess champion in 1991 and 1992) as well as support in the form of unpublished notes from chess programmer Sven Schüle, I will argue that the ICGA's findings were misleading and its ruling lacked any sense of proportion. The purpose of this paper is to defend the reputation of Vasik Rajlich, whose innovative and influential program Rybka was in the vanguard of a mid-decade paradigm change within the computer chess community.

In my view Mark Watkins and Zach Wegner, two members of the ICGA investigative panel who served as technical investigators, provided a series of documents that are accurate in a number of respects. In addition, those on the ICGA panel who supported the finding of plagiarism against Rybka included some rigorously honest and distinguished people. That said, I do quarrel with much of the evidence the ICGA presented and, based on the evidence I will lay out over the course of this paper, I must emphatically reject the conclusion that the similarities between Rybka and Fruit are so overwhelming that Rajlich *must* have copied Fruit code. It is clear that Rybka is an original program by any reasonable standard.

Based on the evidence I will present, a person can form a very credible alternative conclusion: that the implementation of similar evaluation concepts and algorithms in a computer chess program will inherently lead to code similarities even if no code is copied. Thus, the statement "Rybka implements concepts and algorithms learned from Fruit" appears to be the most correct and accurate formulation—a semantically subtle difference that nonetheless completely overturns the ICGA's conclusion. It is important to understand that the implementation of ideas and algorithms learned from other programs is universal practice in chess programming as well as many other types of programming. The issue in contention in this case is whether source code was copied from one program to the next.

Here's the main point: to convict and sentence a man due to his presumed ethical failings and then attempt to ruin him on a world stage you need *a very high standard of evidence*. The ICGA claims its evidence is of a high standard. We shall explore the veracity of that claim.

History of Rybka

On December 4, 2005, the computer chess community was astonished by the initial release of a free, downloadable chess program named Rybka 1.0 Beta, which within days took a sizable lead on all then-existing chess program rankings, surpassing all commercial programs, including renowned engines Shredder, HIARCS, Fritz and Junior. Starting from this top perch, Rybka then proceeded to rapidly widen its lead with subsequent versions. Rybka went on to become a commercial engine in 2006.

Working with Grandmaster Larry Kaufman, one of the world's leading position evaluation specialists, Rajlich issued the seminal Rybka 3 in 2008. Rybka 3 was over 100 Elo points stronger than Rybka 2, an enormous improvement in what was already the leading commercial program. When Rybka 3 was released Rajlich also converted the Rybka 2 series of releases into licensed freeware. In retrospect this was the moment of Rybka's greatest dominance in computer chess: two Rybka editions held the #1 and #2 slots, with the #2 program free, yet stronger than any commercial rival.

The emergence of Rybka meant that affordable personal computers were now capable of exhibiting unprecedented chess playing strength which easily surpassed all human

grandmasters. The implications of this were not lost on professional players, whose preparation methods became more reliant on computers than ever. As a result, Rybka enabled the development of considerable new opening theory and much reassessment of past certitudes. To take just one example, Garry Kasparov's multi-volume *My Great Predecessors* could hardly have been the monumental work it is without pervasive utilization of the best available computer chess software.

Not least among its impacts, though few admit it, Rybka directly or indirectly had the effect of improving its competitors. They learned from Rajlich's innovative search and evaluation techniques which enabled systematic performance tuning, as well as his emphasis on fast searching, as opposed to inefficiently loading up a program with CPU-consuming "chess knowledge". However many view Rajlich's description of his incrementalist program-testing methodology as his single most significant innovation. Dann Corbit, an American programmer and computer chess expert, remarks:

> *…his greatest contribution, the idea that we should test our ideas with thousands of test games to quantify the result, is now copied by literally each and every successful author of a strong chess program.*

Rybka maintained unbroken supremacy on the chess engine rating lists for five years. However its performance in dozens of competitive tournaments held all over the world was, if anything, even more spectacular. Rybka did not merely win nearly every tournament it entered; it won them with a near-90% success rate. It is difficult to overstate the degree of superiority that the Rybka team exhibited in these years in chess software, mastery of hardware, and even in opening theory.

Going from one innovation to another, the Rybka team during this period developed the strongest chess-playing entity in the history of mankind up to the present day: the Rybka Cluster. The cluster, owned by Lukas Cimiotti, a German physician and member of the Rybka team, marshals the power of 300 late-model computer processors and plays chess at a hitherto unmatched and as-yet undefeated level.

Before Rybka it was rare to witness a computer program playing brilliant games with deep, beautiful combinations. Rybka was the first chess program to routinely produce highly artistic masterpieces of chess while avoiding a great many pointless "computer" moves that for many years had been a source of ridicule among strong human players.

For the sake of those not familiar with Rybka's tournament play it might be helpful to study a few of its most stunning games. In the following example the Rybka Cluster demolishes chess program Shredder (winner of 13 WCCC titles) with a series of remarkable moves.

> http://www.chessgames.com/perl/chessgame?gid=1546208

Next, an observer comments on a spectacular positional win over Deep Sjeng:

> *The first part of Rybka's win over Deep Sjeng from 2009 is played like Petrosian on steroids, while the second part is played like nothing I have ever seen in human chess. From my limited human standpoint it looks crazy as Rybka is sacrificing two pawns and allows black to get a strong pawn center apparently without getting much in return. But Rybka (running on 8 cores) judged the position to be a slight advantage for white for most of the game and was relentlessly trying to complicate the position. And complicated it became and when other programs still think Deep Sjeng is much better, Rybka correctly sees that it is winning. According to Rybka Sjeng's*

37. ..Nc7 was a bad move. According to the log file of the game 37...Nb4 would have been much better (score +.48 at depth 22).

http://www.chessgames.com/perl/chessgame?gid=1546726

Finally, another brilliant game wherein the Rybka Cluster on 260 cores defeats chess program Jonny running a super-cluster of 800 Intel i7 cores located at the University of Bayreuth:

http://www.chessgames.com/perl/chessgame?gid=1604306 (also see commentary on http://www.youtube.com/embed/CPFi-LIKNsE)

The latest public edition of Rybka (Rybka 4.1) is more than 400 Elo points stronger than the top competitors that existed in late 2005 on comparable hardware (who, at the time, were themselves approximately as strong as the world's top grandmasters).

While the Rybka Cluster remains unsurpassed in the "unlimited hardware" category and by any standard of strength ought to be considered the reigning world champion of computer chess, a new program for the PC called Houdini, authored by Belgian Robert Houdart, surpassed Deep Rybka 4.1 on the rating lists in December 2010. His latest edition, Houdini 2.0c, is an impressive 60 Elo stronger. Rajlich has indicated that his next edition of Rybka (Rybka 5) will be issued in the first half of 2012. At this point it remains to be seen whether Rybka will retake the top spot among chess programs for the PC.

The ICGA's investigation and outcome

In early 2011 sixteen chess programmers, many of whose programs were in direct competition with Rybka, signed a letter wherein they asserted that Rajlich copied programming code from another engine, Fruit, authored by Fabien Letouzey and released to the public in June 2005, about six months before Rybka 1.0 Beta. They requested that the ICGA investigate their charges and, implicitly, take punitive action on the grounds that Rajlich had violated WCCC tournament rules. At this point over five years had elapsed since the alleged offense, and four consecutive world computer chess championships had been decisively won in head-to-head competition by Rybka.

In response to the accusing letter the ICGA formed a committee consisting of 34 experts, some with genuinely distinguished CVs, for their investigation. Judging from the Wiki which they used during their investigation, approximately seven of these 34 experts actively participated in the discussions. Three of the experts wrote a report wherein they argued that Rybka 1.0 Beta had plagarized large parts of Fruit. None of the actual Rybka versions that participated in the four WCCC tournaments were investigated, although a very close version (Rybka 2.3.2a) was examined following a laborious process of reverse-engineering.

The ICGA committee found that Rybka 1.0 Beta had violated Rule 2 of the ICGA-organized WCCC, and published extensive findings in support of its action.

Ed Schröder was one of the original sixteen signatories calling for an investigation. However, he later began to doubt the ICGA's investigatory process and, ultimately, its findings. Eventually he had a complete change of heart and published his own investigation with the help of five other dissenting chess programmers.

WCCC Rule 2

The basis for the ICGA's action was WCCC Rule 2, related to program originality.

> *Each program must be the original work of the entering developers. Programming teams whose code is derived from or including game-playing code written by others must name all other authors, or the source of such code, in their submission details. Programs which are discovered to be close derivatives of others (e.g., by playing nearly all moves the same), may be declared invalid by the Tournament Director after seeking expert advice. For this purpose a listing of all game-related code running on the system must be available on demand to the Tournament Director.*

Rule 2 has been invoked on a number of occasions. In the 9th World Microcomputer Chess Championship (Portoroz, 1989) the program QUICKSTEP was excluded when it was found to be an unauthorized version of another program. In the 11th WCCC the author of the program LIST was banned from the tournament, and in the 14th WCCC the program LION++ 1.5 was likewise excluded. The case of LION++ 1.5 was discussed by tournament director Jaap den Herik in his article "Interpretation of the Rules" in ICGA Journal Vol 29, p53-54, June 2006.

While no reasonable person can have any quarrel with Rule 2's basic intention to exclude flagrant cheaters the rule nonetheless has some serious flaws which were not clarified in den Herik's 2006 article.

For a non-specialist, including tournament directors and other ICGA board members, the flaws in Rule 2 might not be so obvious. However, to begin to understand the problem with Rule 2 one can start by acknowledging the truth contained within Rajlich's remark on this topic:

> W*hen two modern top chess programs play against each other maybe 95% of the programs are algorithmically the same. What is classing is the other 5%*.

Putting it bluntly, Rule 2 has become obsolete. It is completely vague or unrealistic on critical points that have emerged in recent years, or have always existed but were not as well understood in the past. Years ago, because of the way chess programs were traditionally developed, it was much easier to identify fraudulent entries and programmer-poseurs. Perhaps in that era Rule 2 was quite sufficient to expel entries not meeting originality standards. But, as will be shown, times have changed in computer chess and some of the old standards have been undermined or supplanted due to advances in information technology.

To make Rule 2's absurdity as clear as possible, let me pose some straightforward questions:

- Given the great algorithmic overlap between modern chess programs, what is the definitional distinction between "original" and "non-original" work?

- A modern computer chess program can consist of tens of thousands of lines of code. Which of these lines can a programmer feel certain are in public domain and therefore exempt from Rule 2, and which are not?

- What *exactly* is meant by "game playing code" and on what basis does the ICGA make its distinction?

- What *exactly* is the definition of a "close derivative"? Is this phrase entirely a "we know it when we see it" construct, and if not, then what sensible, consistent, well-defined and articulated principles is such a determination based upon?

- Does Rule 2 *require* all competitors to maintain a copy of any source code they used in competition for an indefinite number of years?

- Can Rule 2 be invoked after tournaments are completed without any time limitation whatsoever? (In law, there is a defense called laches which certainly applies to the Rajlich/Rybka case.)

- Finally, what safeguards exist to prevent *ex post facto* interpretations of rules which are not fully consistent with what competitors understood at the time the tournament took place?

It seems to me rather imperative that a tournament billing itself a "world championship" have crystal-clear rules. These rules should evolve in response to circumstances, contain well-defined procedures and credible enforcement mechanisms, and be designed to protect the integrity of the competition and the title.

A paradigm shift in computer chess

I have mentioned the evolutionary change in computer chess. What I refer to is the fact that the last seven or eight years saw some striking changes in computer chess that were not merely "more of the same". Three extraordinary things happened at the start of that time-span that changed the playing field.

- Information mass-dissemination reached a new level. Computer chess websites where like-minded people could gather and through which useful information could be acquired for free proliferated: computer chess servers, rating lists, discussion fora, testing sites, providers of utilities, blogs, etc. Even something as basic as upload sites have made data transmission much easier than it formerly was.

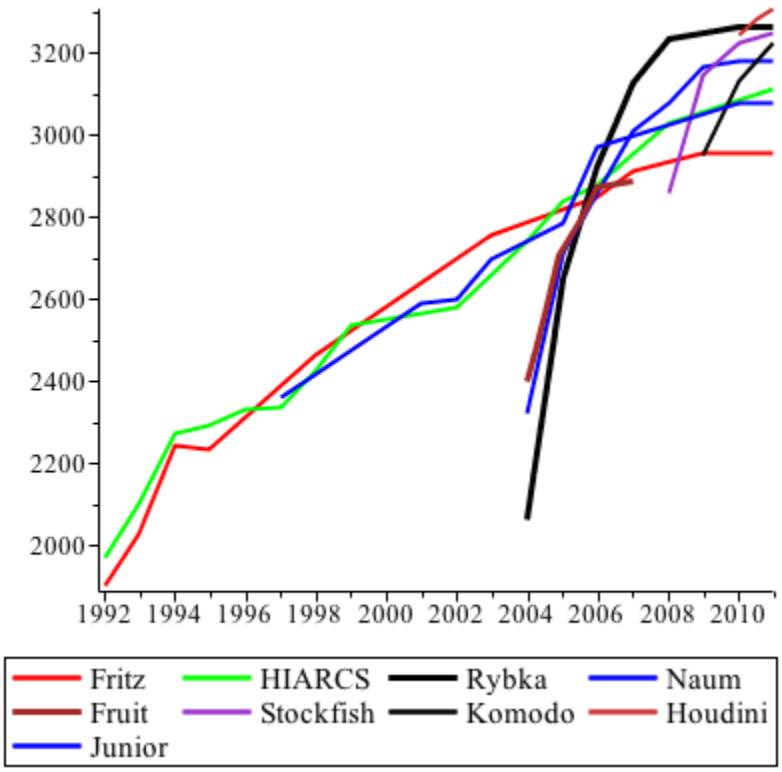
- Hardware steadily improved to the point that around 2004-2005 a capabilities threshold was crossed, and it became possible to test and improve chess programs in ways that were not really feasible on old Pentium IIs and IIIs.

- Most striking, breakthrough program Fruit, authored by Fabien Letouzey, was initially released in 2004 and steadily improved as an open source engine through June 2005.

Taken together, these changes amount to nothing less than what Thomas Kuhn calls a paradigm shift. To illustrate my point we only need to study the patterns in program Elo strength over the past two decades. The following two charts are derived from historical SSDF and CCRL data.

The first chart below shows that chess programs in the past were "slow climbers". Venerable chess programs such as Fritz, HIARCS, Junior, Shredder, etc. progressed steadily and relatively slowly in conjunction with hardware over a long period of time. All top engines from the early 1990s through 2004 fall into this category.

Then something happened in 2004-2005: all new leading chess programs became ultra-fast climbers. Note the slopes of Elo improvement (on the y-axis):

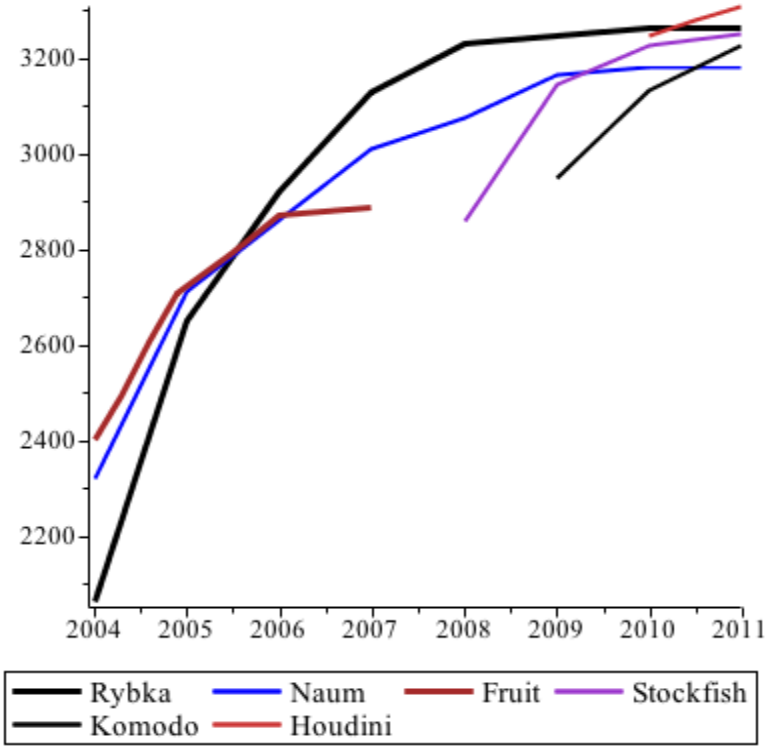


What happened? Starting with the release of the first open-source Fruit in mid-2004, and continuing with the release of subsequent versions of Fruit, open-source engine Stockfish, and *especially* the release of reverse-engineered Rybka derivatives, highly detailed recipes for building strong, modern chess engines have been in the public domain. Fledgling chess programmers as well as programming veterans have not failed to take notice and the state of the art has advanced rapidly. As a result of this spread of knowledge new programs receive a tremendous performance boost and become “fast climbers”.

Robert Houdart, author of current Elo-leader Houdini, provides a short and refreshingly candid acknowledgement confirming this thesis on his website:

> Without many ideas and techniques from the open source chess engines Ippolit *[a Rybka 3-based derivative --SR]* and Stockfish, Houdini would not nearly be as strong as it is now.

Focusing on the last seven years, a number of chess engines either sharply improved around the time that Fruit source code was released, or debuted after Fruit and then soared:



Rajlich comments on the phenomenon:

> *In general top engines always improve in lockstep - the strongest engines are always within 50 to 70 Elo of each other, no matter how fast the top guy is improving. I think this is because the general approach to computer chess is essentially established (unlike say poker), so ideas transplant very naturally from engine to engine. So I think that in 2005 the improvement is probably due to Fruit, between 2005 and 2010 or so the improvement is probably due to Rybka, and most recently it's probably due to Houdini.*

Now, how do the above points relate to the question of program originality?

Well, first of all, a common-sense observation. The charts show how much stronger Rybka was than some of its top rivals such as Fritz and HIARCS. Rybka was as much as 150 Elo ahead of the pack on equal hardware (Rajlich is being modest in his quote above), an extremely impressive achievement and strong evidence that Rybka was substantially original. It had to be, because from first release Rybka was already far ahead of Fruit, and the gap just kept widening:

The next point, less obvious but nonetheless true, is that it is *much* easier to improve the Elo of your own engine if one or more engines exist *that are stronger than yours.* A programmer can tune their engine to play like the stronger engine, which is what many chess programmers have done by playing their engine against a stronger one, sometimes even going so far as to play millions of games in an effort to reverse-engineer the chess knowledge contained within the stronger engine.

Rybka 4 x64	3128
Rybka 3 x64	3094
Rybka 3 w32	3048
Rybka 2.1o x64	3012
Rybka 1.2f w32	2927
Rybka 1.1 w32	2892
Rybka 1.0 Beta x64	**2868**
Rybka 1.0 Beta w32	2816
Fruit 2.1 w32	**2712**
Crafty 17.14	2475

Elo-ratings are for 1-CPU hardware (CEGT data)

It is *much* more difficult to improve your engine at the same rate when you are far ahead of the field because you do not have a stronger engine to emulate or measure yourself against. Yet Rajlich managed to consistently do exactly this with Rybka between 2005 and 2010. The amount of original, path-breaking work that *must* have occurred to achieve this feat is truly staggering.

Next, note that even a "slow climber" like HIARCS had its biggest Elo jump in 12 years just six months after the Fruit 2.1 source code was released (in version HIARCS 10). Multiple WCCC-winning Junior made an even more dramatic improvement after the release of Fruit. For that matter, not a single new (and serious) program that came after the release of Fruit followed the old-paradigm Elo growth curve.

However, these facts do not merely suggest that everyone in the top tier of chess programming learned from Fruit. Retrospectively, what now seems clear is that Fruit also unwittingly triggered a revolution in the whole ethos of chess programming. From the emergence of Fruit and going forward, the premise within the programming community was that it was perfectly fine to re-use and share ideas and algorithms from leading programs *whether they were open source or not*. This was a new attitude which led us to the current landscape in computer chess.

With hindsight, it seems the ICGA did not recognize the transformation that took place in computer chess following the release of Fruit. For failing to recognize the change and its implications in 2004-2005 the ICGA can certainly be forgiven; institutions almost never immediately see when long-established traditions and norms have quietly slipped into obsolescence. But for them to *still* not recognize in 2011 how the chess software development panorama had changed, and to fail to adapt accordingly, risked putting the integrity of the WCCC enterprise in jeopardy, and indirectly resulted in the venerable organization being led astray.

How to succeed in programming without really trying

While Fruit's release was the first indication that the game was changing, it was at first not plainly obvious to everyone that Fruit was revolutionary. Initially, the only ones who recognized it for what it was were chess programmers, who over a period of time learned from the program and adapted ideas in bits and pieces to their own designs.

What made Fruit such an epochal breakthrough was not really new ideas in evaluation or search, two of the most critical components of a chess engine. There were improvements in these areas to be sure but they were relatively incidental. The thing Letouzey accomplished, which changed computer chess forever, was to express old ideas far more cleanly and efficiently than anyone else had done before. Fruit had a visionary program structure, and this stride forward in 2004-2005 was made all the more significant by his choice to make the program open source.

What happened next is a tale many in the hobby know well. Rybka was released six months later incorporating every useful Fruit idea that could be incorporated, and adding many more improvements. As I have established, Rybka assumed a position of huge dominance. Other program authors likewise learned from Fruit and went after Rajlich like a pack of hounds chasing a fox.

But then something unprecedented in computer chess happened. In May 2007 a strong closed source program called Strelka (literally "arrow" in Russian, but a reference to a Soviet canine that orbited the Earth) was released by a hitherto unknown individual going by the name 'Jury Osipov'.

This program was immediately thought to be a very close derivative of Rybka because its solving of test positions was extremely similar. But, beyond the objective measures of similarity testing, Strelka had to have been a reverse-engineered Rybka derivative because, at the time, a new program of such strength and manifest similarity in its playing style could hardly have come from anywhere else. Thus a very public precedent was established: someone had reverse-engineered a closed-source program with impunity.

The next step on this path came in January 2008 when Strelka 2.0 was released, this time with open source code. In other words, Rybka 1.0 Beta's secrets were now revealed for any programmer to use. Is any reader gullible enough to think the community of chess programmers took the moral high ground at this point, condemned 'Osipov', or refused to even take a look at Strelka's source code?

Sometime thereafter someone with larcenous intent approached Rajlich via email posing as GM Larry Kaufman. The result of that caper was that another key Rybka secret, namely how Rajlich tuned his evaluation, was inadvertently revealed to dishonest individuals whose apparent goal was to forcibly strip every last bit of proprietary information from Rajlich.

Fast forward to mid-2009. Rybka 3 was reverse-engineered by a group of chess programmers using fake names (Yakov Petrovich Golyadkin, Igor Igorovich Igoronov and Roberto Pescatore, i.e. an Italian rendition of Bobby Fischer). This effort was published on the Internet as the IPPOLIT program, and soon thereafter chess programmers were freely using this source code to create strong Rybka-flavored derivatives, such as Robbolito and IvanHoe.

In a Talkchess forum post sent to Sven Schüle, a German chess program author, Rajlich wrote:

> *Ippolit is disassembled Rybka 3 with changes. The changes are considerable but not even close to enough to leave any doubt. Robbolito is an evolved Ippolit, with more changes and more cleanup. I haven't checked the other new engines yet.*

Initially, some competing programmers expressed qualms about using code produced by fake-named authors. But apparently the temptation became too great and the ideas were taken.

I recite this well-known history to provide the reader with background context. What is clear is that Rajlich's original ideas have been lifted from various reverse-engineered editions of Rybka again and again and again--his work has been pilfered as comprehensively as anyone's in all of computer chess history. A number of leading programs have rather obviously benefited: in practical improvements to their programs, in tremendously reducing their development time, and in gaining insights they might never have developed on their own. Yet, it is Rajlich who was investigated and found guilty of plagiarism *in absentia*, banned for life, stripped of titles, and vilified in the international press over a five-year-old tournament rule violation.

Ironically, it is the ICGA which may have committed an illegal act in publishing a disassembly of Rybka 2.3.2a, a commercial product still under license, on the Internet:

> Article 6 of the 1991 EU Computer Programs Directive allows reverse engineering for the purposes of interoperability, but prohibits it for the purposes of creating a competing product, and also *prohibits the public release of information obtained through reverse engineering of software.*

On the matter of "plagiarism"

The first thing that must be addressed regarding the ICGA's finding that WCCC Rule 2 was violated is the formal language they use in their allegation. They specifically accuse Rajlich of "plagiarism". I note that the writers of this charge are scholars with doctoral degrees and are men whose profession obliges them to choose their public words with careful precision.

Consequently an important distinction must be made that is more than merely semantics: Rajlich provably did *not* commit the dictionary definition of plagiarism:

> *The unauthorized use or close imitation of the language and thoughts of another author and the representation of them as one's own original work, as by not crediting the author.*

From the initial release of Rybka 1.0 Beta until early 2011 the general consensus in the computer chess community was that Rybka was an original program, even though it had undoubtedly learned a great deal from the leading open source programs in existence in 2005, Fruit and Crafty. This consensus was based on Rybka's position on the Elo rating lists and steadily increasing strength, its search behavior, unique features, etc.

Rajlich conceded from the very beginning that he had studied Fruit's open source code very closely and learned a great deal from it. In an interview from 2005, right after Rybka 1.0 Beta was released, Rajlich acknowledged the computer chess community's as well as his own debt to Fruit:

> *Yes, the publication of Fruit 2.1 was huge. Look at how many engines took a massive jump in its wake: Rybka, HIARCS, Fritz, Zappa, Spike, List, and so on. I went through the Fruit 2.1 source code forwards and backwards and took many things.*

Rajlich later publicly praised the work of Fabien Letouzey:

> *I don't want to get too specific about which ideas from Fruit I think are really useful, but they fall into two categories:*
>
> *1) Very specific tricks, mostly related to search.*
> *2) Philosophy of the engine (and in particular of the search).*
>
> *Fruit could really hardly be more useful along both of these dimensions. Fabien is a very good engineer, and also has a very clear and simple conception of how his search should behave.*

Finally, Rybka 1.0 Beta's Readme file gave credit to Fruit in a "Special Thanks" section:

> *…for Fruit, which shattered a number of computer chess myths, demonstrated several interesting ideas, and made even the densest of us aware of fail-low pruning.*

By definition, plagiarism ***only*** happens when credit to sources is ***not*** given, which was *never* the case with Rybka. These acknowledgements were widely known at the time Rybka entered

WCCC tournaments and, rather obviously, Rybka's debt to Fruit steadily diminished over the years as program improvements superseded the original program code in Rybka 1.0 Beta. If these acknowledgements were insufficient to satisfy WCCC Rule 2, then we must return again to arguments made earlier on the inadequacies of the rule itself and how it is applied.

If plagiarism did not actually take place, then what is the actual nature of the ICGA's complaint? Answer: the ICGA has no complaint at all unless it directly and without equivocation asserts that Rajlich copied original source code and used the resultant chess program in a WCCC tournament competition.

Whether or not he speaks for the ICGA, code-copying is precisely what leading ICGA panelist and secretariat member Dr. Robert Hyatt has made dozens (if not hundreds) of times in the Rybka forum. While Hyatt is not officially an ICGA spokesman (only Dr. David Levy can speak for that organization), he did lead the Rybka investigation, he wrote or supervised the ICGA report, he energetically and persuasively presented his arguments to other panel members, and in the time since he has made several thousand online posts in defense of what was done. A discussion of the ICGA report can scarcely be conducted without extensive reference to his views on the case against Rajlich. Accordingly I will develop the Rajlich defense under the assumption that Hyatt speaks for the ICGA.

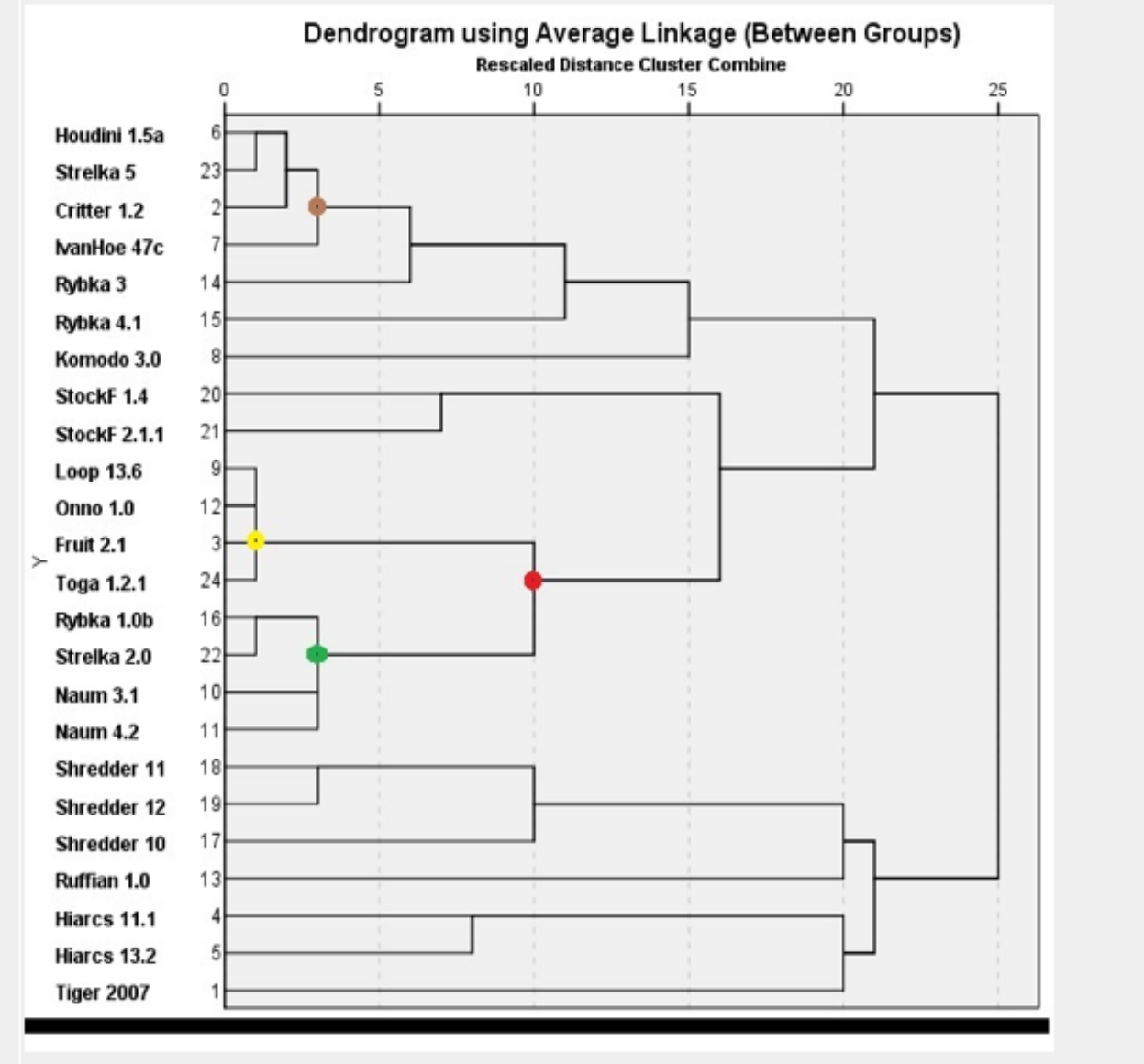
Playing similarity testing of computer chess programs

I will start my rebuttal to the ICGA's technical findings by addressing a subject that is *not* addressed in the ICGA report, but ought to have been.

When comparing two chess engines recreationally what really matters to us as hobbyists and what makes it all so engrossing is how *differently* they play. On the other hand, when comparing chess engines in a code-copying investigation what really matters is how *similarly* they play. After all, the sole function of a chess program is to play chess. Would you think it acceptable if two chess programs played *exactly* the same moves while having completely different program code? Obviously not, as this would be *prima facie* evidence of a direct translation of program A into program B, rather than an original program.

To put matters into better perspective let's consider the chess evaluation similarities between Fruit 2.1 and Rybka 1.0 Beta compared to the similarities between other chess engines.

In an interview with Nelson Hernandez (http://www.youtube.com/embed/cQshTNJ4pSM) Rajlich suggested that one measure of originality and similarity of chess engines can be quantified by considering ponder-hit rates (the percentage of the time two programs will come up with identical moves in a large number of selected positions). Kai Laskos, a computer chess theorist with statistical expertise who posts in the Talkchess computer chess forum, took up the challenge and performed a statistical analysis of ponder-hits using a similarity-testing program written by chess programmer Don Dailey. This analysis produced statistical "fingerprints" of a number of chess engines. Then, employing a hierarchical clustering algorithm, he produced a number of dendrograms (using IBM SPSS) that elicited some surprises but also confirmed what was already known. Bear in mind Rajlich had no idea a rigorous analysis would actually be performed when he was interviewed by Hernandez; the results could easily have proved him a liar.



Dendrogram comparing ponder-hit similarity of numerous chess engines. Thanks to Kai Laskos for creating this chart and posting it to the Talkchess forum

The way to read the chart is as follows:

- The closer to the left edge the programs diverge means the more similar they are.
- The red circle is where Rybka diverged from Fruit.
- The green circle is where Naum diverged from Rybka/Strelka.
- The brown circle is where IvanHoe 47c diverted from Rybka 3.
- The yellow circle is where chess programs Loop and Onno diverged from Toga.

Laskos points out that Junior 12.5 (not shown) appears unrelated to anything, and that relationships at a large distance from the left edge (~20) are most likely statistical noise. Note that Strelka 2.0 is an admitted clone of Rybka 1.0 Beta; you can see that by how very close to the left they diverge.

Three distinct clusters of derivative engines can be identified:

- Programs that are Fruit derivatives (children of the yellow circle): note that Rybka 1.0 Beta is not one of these

- Programs that are Stelka 2.0 derivatives, which is a Rybka 1.0 Beta deriivative that was obtained through reverse engineering (children of the green circle)

- The Strelka/Ippolit/Robbolito/Ivanhoe/Houdini program cluster (children of the brown circle, not all of which appear in the dendogram), which trace their origins to a reverse-engineered version of Rybka 3

The ponder-hit "fingerprints" show that the similarity between Fruit and Rybka 1.0 is actually not that large, and certainly not when compared to other programs. Moreover, the difference between Fruit and later versions of Rybka 3 and Rybka 4.1 is so large (~20) that it's fair to say there is no similarity at all in practical terms.

Given the data-set used by Laskos the probability of "over-fitting" in his chart seems low, and thus his results appear statistically significant. The similarity fingerprints suggest that Rybka 1.0 Beta was actually remarkably "clean", and much more so than numerous engines that have emerged in the years since. Note the irony: the chess program that has been the most influential to the others over the past several years (for very good reasons) is the target of plagiarism allegations!

Dr. Miguel Ballicora, author of chess program Gaviota, came up with similar findings based a larger set of data. A large set of program "trees" is vitally important to identify program similarity, because with more data 'families' of programs emerge more clearly. Ballicora found that the relationship between Fruit and Rybka is NOT significant. With more data you soon realize that the distance between Rybka 1.0 Beta and Fruit 2.1 is *not* less than other unrelated engines.

In no way do I mean to suggest that the approach taken by Dailey and Ballicora is necessarily the best or only way to objectively measure program similarity. Ponder-hit analysis is not perfect and it may be possible to fool the methods that have been advanced. But ponder-hit analysis does provide one measure of value that, combined with other techniques, could result in a robust suite of originality tests. It is a promising methodology simply because it is based on hard numerical data and methods that can be applied to all programs in the same way. Additionally, it has the benefit of being conceptually easy to understand. In time, improvements to this approach will undoubtedly emerge and, hopefully, settle the most visible and relevant basis for comparison: how a chess program actually plays.

What defines an original program?

The widely understood 'originality' standard among programmers is that they may adopt the ideas of others but not duplicate their actual source code. The problem that has emerged with this construct, and intensified in recent years, is that many of today's programs have been influenced by the work of other programmers, and sometimes heavily. Theoretically this could have happened in such a way that one program that has directly or semantically copied portions of source code could easily be more dissimilar to its unacknowledged source in actual play than

another program, which copied nothing, but rigorously applied and then obfuscated ideas and algorithms taken from the same source.

An aspect of the originality problem was summarized by Fabien Letouzey in a 2008 interview:

> *I am sorry to say that whether an engine is someone's "own work" makes little sense to me, although I understand that tournament directors would like a clear yes or no.*
>
> *The reason is that all engines, whether amateur or commercial, share most of the techniques. Alpha-beta (of which PVS, NegaScout and MTD(f) are only derivatives), iterative deepening, check extensions, null move, etc ... are shared by most and have been published, mostly by researchers, some of them more than 30 years ago! Sure there are many different ways to represent the board and pieces but it only affects speed, which rarely amounts to more than a few dozen Elo Points.*

The ICGA's core finding of code similarity relates to Rybka's evaluation function, which it is asserted came from Fruit. But from where did Fruit's own evaluation function originate? Letouzey explains in the same interview that Fruit's positional evaluation function is itself not, strictly speaking, original:

> *I can't think of a search feature in it that was not described before. Ditto for evaluation terms (except perhaps a few drawish-endgame rules that activate in one game in a hundred).*

He continues:

> *Unfortunately my interest in working on evaluation is very low because I don't learn anything in the process and I cannot apply the same changes to games other than chess. For this reason I cannot promise important future development on Fruit.*

The ICGA alleges there is a great similarity between Fruit and Rybka 1.0 Beta evaluations but, as I will show next, all their analysis actually demonstrates is that Fruit and Rybka use similar algorithms for their evaluation. Algorithms, let us be clear, are expressions of ideas, *not* source code. And even if they *were* source code it is unclear whether such things as evaluation terms (i.e. positional features that are rewarded or penalized in the evaluation, such as doubled or passed pawns, mobility, king safety, etc.) actually count as "game-playing code" as WCCC Rule 2 is currently written.

Evaluation: a tale of two programs

The core essence of the ICGA's case against Rajlich is that Rybka and Fruit have very similar positional evaluations, i.e. a chess program's mathematical assessment of which side is winning in a given position. Dr. Hyatt announces in the Rybka forum that he is so confident that Rajlich has been so completely and redundantly busted in this area that no further analysis is needed:

> *I suppose one could have gone into every bit and piece of Rybka to see what was original and what was copied, but after the evaluation study, there seemed to be little justification for the effort required...*

The problem is that the ICGA's findings on the evaluation similarities between Fruit and Rybka tend to fall into one of these three descriptions.

1. The evidence presented is untrue.

2. The evidence is true but the conclusions are false and/or tendentious.
3. The evidence is true and the conclusions are true, but put into proper context, the conclusions are irrelevant or immaterial.

I'll go over some of the ICGA's specific charges in due course, but first it is important to itemize the significant ways Rybka's evaluation differs from Fruit's.

The first big evaluation difference between the programs is that they are grounded on different valuation conventions. Fruit evaluation is based on a programming tradition going back decades which stipulates that a pawn is worth 1.00. From its initial release Rybka's evaluation was based not on piece values but projected winning percentages. Per Rajlich, this mathematically subtle difference plays a significant role in testing.

Ed Schröder reveals five additional major differences versus Fruit in his investigations:

1. *"Lazy" evaluation is not in Fruit but is present in Rybka.*

2. *The programs have entirely different futility pruning approaches.*

3. *Fruit has only one evaluation array related to King Safety. Rybka has many.*

4. *The Fruit "quad" function calculates a value based on the rank of a passed pawn in an unusual way, using up valuable processor time when this can be done in just one instruction via a PST or rank-based table, as is done in Rybka and many other programs.*

5. *Fruit evaluates in two steps, while Rybka directly adds up an evaluation score.*

That makes six differences altogether: differences that are actually substantial and impact playing strength, unlike some of the tangential issues discussed in the ICGA report. But next Rajlich points out four *more* big differences in this illuminating passage:

> *Fabien was a big search guy who basically didn't care about eval, but he nevertheless really hit on a major point with his eval. His eval was based completely on mobility. This has two nice properties. The first is that it interacts well with the search - having the right to make more moves tends to coincide with the chances that more searching will improve the score. The second is that it's symmetrical and continuous, because all pieces are basically handled in a similar way. Every piece has the right to move to certain squares. This symmetry is elegant, eliminates discontinuities, makes the eval smooth, helps with tuning, etc.*
>
> *Re. Rybka 1.0 Beta vs Fruit 2.1 eval - I don't know the exact differences. There must be a lot of little things. Generally I would say that Rybka 1.0 Beta had the following big eval innovations:*
>
> *1) Material imbalances - Rybka was the first engine to understand that major pieces are more valuable in endgames. See for example the game Ikarus-Rybka from Paderborn 2005 - every engine except for Rybka thought that white was much better in that NN vs RP endgame, while any decent player would know that black is perfectly fine.*
>
> *2) Passed pawns - passers are the other major exception to mobility based evaluation. Rybka 1.0 Beta had quite a few heuristics for scoring passers, I am quite sure again that these are far ahead of Fruit or other engines from 2005.*
>
> *3) Tuning - I had my own eval tuner which was kind of primitive compared to what I have now, but nevertheless I think that it was better than what Fabien and others had in 2005.*

These are worth maybe 20 Elo each.

One other unusual feature of Rybka's eval from that time is that I tried to have as few correlated eval terms as possible. I took this pretty far. For example Rybka didn't score doubled pawns (I'm not sure about the exact versions but I think this applies to Rybka 1.0 Beta). A doubled pawn penalty is mostly redundant with penalties for a weakened king shelter or for the inability to create a passed pawn, so Rybka would only score the underlying issues (i.e. the king shelter and passed pawn creation). I later decided that this was wrong, but anyway it's a unique feature of Rybka's eval from 2005.

The ten substantive evaluation differences outlined above, combined with Rybka's entirely different search and board representation, signify that Rybka and Fruit must be considered two different chess engines by any reasonable person. These differences go a considerable distance to explain why, per every independent rating group, the 64-bit version of Rybka 1.0 Beta played some 150+ Elo points stronger than Fruit 2.1 (which only came in a 32-bit version).

We can see that Rajlich's evaluation was materially different from Letouzey's in at least ten ways, but how did he develop these ideas? We find our answer in new information about Rajlich's early programming R&D work. During 2004 and 2005 Rajlich wrote himself a series of notes, some of substantial length, on evaluation. He kindly emailed me some of these notes which make it amply clear that he was intellectually engaged in evaluation and that copying was the furthest thing from his mind.

These files were written in the same period that his accusers claim he spent feverishly copying Fruit's evaluation. It is exceedingly hard to see the point of developing a slew of original ideas for Rybka only then to copy Fruit's evaluation.

Name	Date modified	Type	Size
Evaluation (3.8.05).doc	03/08/2005 17:01	DOC File	212 KB
Evaluation Components (28.6.04).doc	03/08/2005 12:20	DOC File	26 KB
Evaluation Testing - original ideas.doc	03/02/2005 13:22	DOC File	74 KB
Evaluation Uncertainty (16.3.05).doc	16/03/2005 20:53	DOC File	51 KB
Framework Components and Testing.doc	31/05/2005 21:58	DOC File	54 KB
Material Imbalance Valuation Summary.doc	22/03/2003 20:39	DOC File	152 KB
Position Evaluation.doc	16/11/2003 06:42	DOC File	36 KB
Positional and Tactical Search.doc	05/02/2004 17:02	DOC File	31 KB
Progress Notes 04.06.16 (Testing Methodology).doc	30/06/2004 11:24	DOC File	29 KB
Quiescence Search and Evaluation Uncertainty.doc	09/06/2005 18:41	DOC File	70 KB
Root Node Pre-Processing.doc	07/12/2004 18:56	DOC File	91 KB
Sanity Testing.doc	09/06/2005 15:02	DOC File	33 KB
Search Framework.doc	15/12/2004 18:04	DOC File	76 KB
Search Ideas.doc	08/02/2004 17:24	DOC File	34 KB
Search Literature Review.doc	21/07/2005 11:44	DOC File	58 KB
Testing Algorithm.doc	21/12/2004 09:36	DOC File	49 KB
xUCI Specification.doc	19/08/2005 21:33	DOC File	25 KB

There is another difference between Rybka and Fruit that merits comment. A common misperception following the ICGA's report was that Rybka transcribed Fruit's evaluation practically verbatim into a different board representation, and was principally different from Fruit in "the search" (i.e. algorithms related to searching for the best move). As we have already seen, the idea that Rybka's evaluation is the same as Fruit's is totally wrong both in the specifics and the underlying premise.

To further clarify this point, Fruit used a "mail-box" representation of the chessboard, while Rybka used a "bit-board" representation. How a chessboard is represented in a program has nothing to do with evaluation; it is purely a difference in program architecture. Rajlich dismisses the importance of chessboard representation with this comment:

> *If you take Fruit's evaluation and modify it from Fruit's board representation (called mail-box) to Rybka's board representation (called bit-board) no serious Elo difference is expected except possibly slightly lower Elo on 32-bit processors and slightly higher Elo on 64-bit processors.*

Given the points I've outlined above what are we to make of the following categorical statements made by Zach Wegner in his ICGA report findings?

> *Simply put, Rybka's evaluation is virtually identical to Fruit's*
>
> *Overall, the pawn evaluations of each program are essentially identical.*
>
> *Because of Fruit's unique PST initialization code, the origin of Rybka's PSTs in Fruit is clear.*

These are all demonstrably incorrect and tendentious conclusions which would be extremely misleading to someone who lacked the requisite technical expertise or was not prepared to invest the necessary time to study the full contents of his paper.

Feature Overlap: garbage in, garbage out

Mark Watkins, in his analysis of Rybka-Fruit similarities, compares several chess engines with respect to their evaluation "features" and shows that Rybka 1.0 Beta has an "eval feature overlap" with Fruit 2.1 of about 74% (Rybka 2.3.2a is judged to be about 64%).

Watkins shows "feature overlap", not a "code overlap". The precise definition of an evaluation term in a chess engine (e.g. "rook on the seventh rank") is a mathematical *formula* which is calculated by an *algorithm*. The algorithm itself is an abstract concept. It is implemented in a programming language based on explicit data structures defined by the surrounding program—that is called "code". But Watkins' evaluation feature is in actuality the formula expressed by an algorithm. This formula is on the *conceptual* level and therefore, according to accepted practice, everyone is free to use it. Thus his entire analysis lacks traction.

But there are a few points that ought to be made about his analysis. First, his choice of engines to compare against Rybka and Fruit are relatively weak, and this fact puts the practical value of their evaluation feature set in comparison to world-class engines under question.

Next, it should be mentioned that the assignment of "feature overlap" values for each single evaluation term, using a scale of 0.0 to 1.0, was based on inherently subjective judgments. Given Watkins' analysis template, if we were to ask a group of programming experts to assign overlap values to the engine pairs under question, one cannot be sure how close Watkins'

values would be to the average values they assigned, let alone how closely his values would correspond to practical reality, which in any event is impossible to calculate with precision.

Finally, there is the matter of data interpretation. Even if we ignore the points cited above, questions must be raised. Why is an overlap value in the range of 40%-44% "allowed" but a value of 64% (Rybka 2.3.2a vs. Fruit) or 74% (Rybka 1.0 Beta vs. Fruit) "not allowed"? Who sets these standards and what are they based upon? Can the ICGA create and enforce new standards years after a tournament is completed?

Dots amazing: the case of the errant '0.0'

A source of strife since the ICGA issued its report has been an analysis of Rybka's time management code. All chess engines have to ration how long they can spend thinking about a move based in part on how much time they have left on the chess clock. Time management, obviously, is as important in computer chess as it is in human chess, particularly in situations when time remaining is down to a few seconds.

Ironically, the basis of the ICGA's argument boils down to an interpretation of one line of source code in Rybka 1.0 Beta which they *believe* contains '0.0'. (No joke, '0.0' appearing in a program written in 2005 has been a *major* issue for the ICGA investigators.)

Fruit used a system of floating point numbers or "floats" (such as "0.0") for managing its time. Rybka 1.0 Beta had a faster and simpler approach using integers (such as "0") for checking time.

There is a time check within Rybka 1.0 Beta that the ICGA investigation team says looks like this:

> If (movetime >= 0.0)

There is a time check in Fruit the looks like this:

> If (movetime >= 0.0)

So the ICGA investigators argued the following:

> Rybka uses integer based time management so we would expect Rybka to look like this:
>
> If (movetime >= 0)
>
> The fact that Rybka does not utilize an integer format, and instead uses a floating point convention just like Fruit, is undeniable proof that code-copying occurred.

I asked Rajlich how the '0.0' might have happened in Rybka and this was his response:

> *I don't know where the 0.0 came from. It's definitely weird/wrong. Rybka was UCI from the beginning, even back when everybody was using WinBoard. I would say that every two to three years I do a big cleanup of this code. This might take a few hours, and then I won't touch it until the next time. My first UCI parser actually used inheritance, I was extending UCI to do some testing, but that was gone even before Rybka 1.*

This entire line of argument started years ago on Talkchess with a post by Rick Fadden, wherein he pointed out the floating-point versus integer format mismatch. This observation, which I have no reason to think was not made in good faith, was probably the public origin of the Rajlich controversy. I say this because this piece of seemingly concrete evidence placed into the psyches of rival chess programmers that Rajlich *must* have copied code from Fruit, and once that was accomplished all that was needed was someone like Fabien Letouzey to return from computer chess retirement to light the fuse.

But here's the thing: Fadden assumed that Rajlich really typed or copied '0.0'. It is quite possible that his assumption was incorrect. Remember, Fadden didn't have Rajlich's original source code either; his output only indicated that *something* extraneous to integer format was on that line of code.

Rajlich could have typed this instead:

```
If (movetime >= 0.)
```

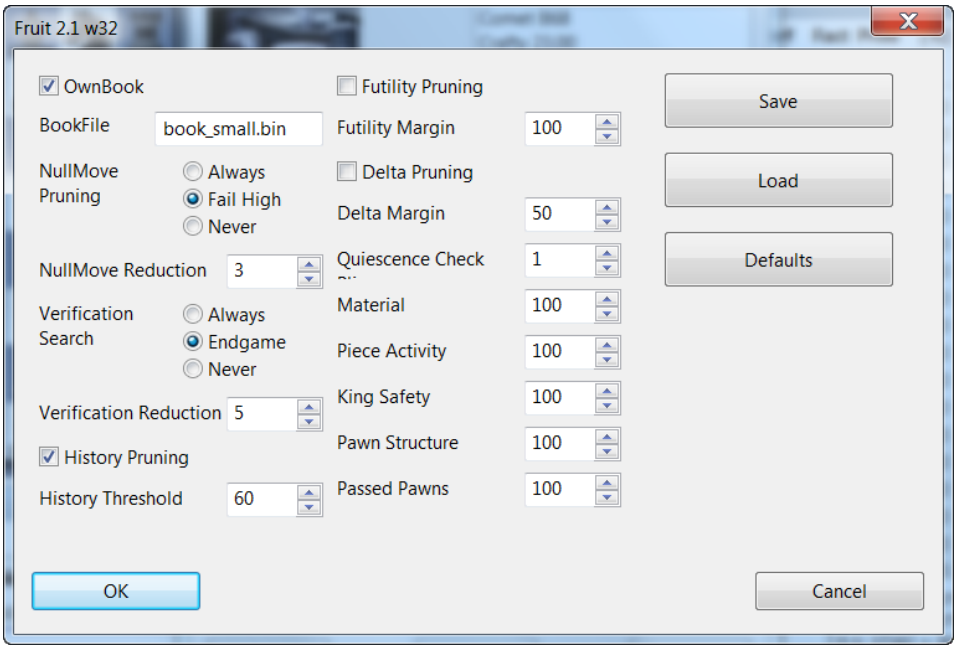
In other words, he could have just added a *dot* to the zero. If he did, this would have compiled to exactly the same floating point compare instructions as if '0.0' had been coded.

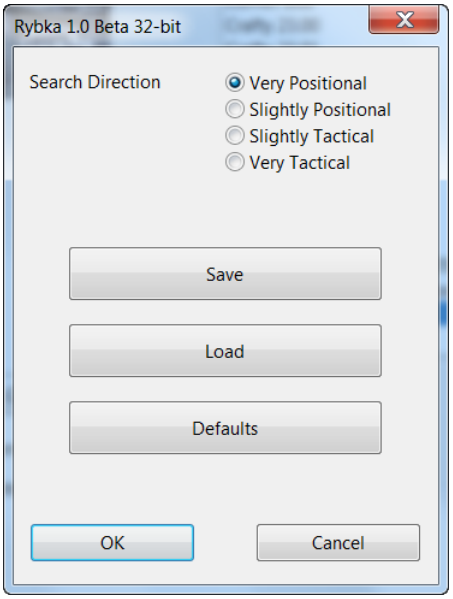
The technical experts who helped me write this paper could scarcely believe this point when it first dawned on them. They researched and double-checked, and found that on Microsoft compilers contemporaneous with 2005 this observation is indisputably correct.

The truth of the matter is that there is no definitive and provable answer. I came to the conclusion that '0.0' is a litmus test. If you believe that Rajlich is guilty of code-copying then '0.0' reinforces that belief and is your smoking gun. If you believe that Rajlich is innocent then you are apt to conclude that typing '0.' (not '0.0') was a simple coding oversight. Further mitigating circumstances I can offer to those in the guilty camp are these:

1. Time management is not "game-playing code" (per ICGA Rule 2). It is interface code from the engine to the outside world, i.e. Rajlich's reference to a "UCI parser".

2. Comparing the UCI parameters for the two engines reveals they are markedly different just as we saw with the comparison of Rybka and Fruit evaluations. Fruit 2.1 has twenty configurable UCI parameters (hash-size not shown in the figure below). Rybka 1.0 Beta, in contrast, has only two such parameters.





But ultimately the “big picture” argument is the most compelling. This contentious ‘0.0’ issue comes down to a dispute about one extra keystroke, one single dot, on one line of code that has zero impact on how the program actually plays. On what reasonable basis can a person conclude from this one superfluous dot that Rybka is non-original and Rajlich deserves to have all his titles stripped and be banned for life? How could this literally nugatory piece of evidence tip the scales in favor of the prosecution? How many devils can dance on a dot of code?

Piece-Square Tables: sound and fury signifying something

One of the central arguments in the ICGA report has been that Rybka and Fruit had very similar “Piece-Square Tables” (PSTs). PSTs are numerical values within eight-by-eight arrays representing a chessboard, with each array corresponding to a different chess-piece, usually

expressed as integers and arranged into simple symmetrical patterns. The purpose of PSTs is to modify a position's evaluation based on the square location of various chess pieces.

Rajlich comments on the practical importance of PSTs:

> *The effect of PSTs is minimal but probably positive. Any reasonable choice of PST values leads to +/- less than 1 Elo.*

Rajlich's comment is notable for both its brevity and its significance. The first realization one makes is that the whole PST case in actual playing terms is a trivial sideshow whatever the merits of the ICGA's assertions. PSTs, per Rajlich, have no material impact on playing strength and possibly no *measurable* impact within standard statistical confidence levels. Rajlich is evidently not even sure himself based on his remark that PSTs are "probably" a positive factor.

To be efficient the PSTs should use at most two or three CPU cycles. This explains why Fruit and Rybka evaluations were based on PSTs with simple integer patterns (e.g. +1, +0, -1, -3); integers are quicker than floating-point numbers. None of the PSTs had been optimized in Rybka 1.0 Beta, hence their simplicity. Both Fruit and Rybka 1.0 Beta use PSTs that were generated through the use of simple formulas that reflected then-common chess knowledge. Even though Rajlich recognized that PSTs had only a tiny impact he still wanted to have the ability to fine-tune his evaluation. There was no demonstrated intent to obfuscate Fruit integers in Rybka's code as is stated in the ICGA report.

In actual fact, *every one* of Rybka's PST values is different from those in Fruit. Dr. Miguel Ballicora persuasively shows here and also here that Rybka and Fruit PST tables are *totally* different. In addition, Ed Schröder has summarized Ballicora's and Chris Whittington's analysis on his website here. I will not recapitulate the arguments of these gentlemen here, as they are quite technical, but let me suggest that they demolish the PST case.

Based on the findings at the three linked sites above we could stop discussing PSTs right here and move on to the next topic, but Dr. Hyatt and his supporters have invested a great deal of effort defending PST as definitive evidence of "code-copying". In the process, they dug an ever-deeper hole for themselves and, I regret to say, have not stopped their excavation work.

We must press on with the topic of PSTs because it puts the whole problem with the ICGA report in the spotlight. When I, like many people, first read the ICGA report I found their tables of virtually identical, side-by-side Fruit and Rybka code highly incriminating. This was surely the same reaction ICGA head Dr. David Levy had, as well as panel members not serving on the report-writing "secretariat". Given the evidence that was presented, *and assuming its veracity*, the final outcome of the ICGA investigation was an absolute and undisputable certainty. On this point we need to be very clear.

It really goes without saying that the panel members voted based on the findings of the ICGA report, and it would have been extremely prejudicial to the whole process if the report presented data in a misleading way.

Let's start unraveling the PST mystery by reading what Rajlich said to Dr. Levy in a terse email when the ICGA investigation was in progress:

> *I'm not really sure what to say. The Rybka source code is original. I used lots of ideas from Fruit, as I have mentioned many times. Both Fruit and Rybka also use all sorts of common computer chess ideas.*
>
> *Aside from that, this document is horribly bogus. All that "Rybka code" isn't Rybka code, it's just someone's imagination.*

Rajlich issues a flat denial. What is characterized as “Rybka code” in the report, he says, is “horribly bogus”. You can only come to one conclusion: either Rajlich is flat-out lying or the ICGA report is wrong, or possibly even fraudulent. There is no middle ground.

The ICGA report contends that Rybka used Fruit’s PSTs in its evaluation function. To support this the report provides page after page of near-identical source code side by side. However, it turns out on closer inspection that the most damning portion of the ICGA report is in fact *a work of fiction*, and as a consequence the parts of the ICGA report related to code-copying are profoundly misleading. Ed Schröder’s reports [1] [2] make the case powerfully.

One of the first insights that led to the revelation that the PST “Rybka code” might come from another source came from one of the chief ICGA accusers, chess programmer Mark Lefler. Lefler nonchalantly pointed out in an online post that "[Rajlich] could have used a spreadsheet" for PST generation, an admission that undermined the sourcing and importance of PSTs.

But this crack in the ICGA argument was a trifle compared to the subsequent train of events. Critics of the ICGA soon realized that *no one* has actual Rybka source code from before 2010, not even Rajlich himself, who sheepishly admitted to Nelson Hernandez off-camera in the course of their July 2011 video interview that he had never maintained any form of version control for Rybka source code until Rybka 4.

This jaw-dropping admission is entirely believable because Rajlich asked for copies of his own program long before the ICGA controversy started. In mid-2010 he makes this rather embarrassing request in the Rybka forum:

> *Can someone please post here all of the Rybka 2.3.2a versions? (I don't seem to have a copy any more.).*

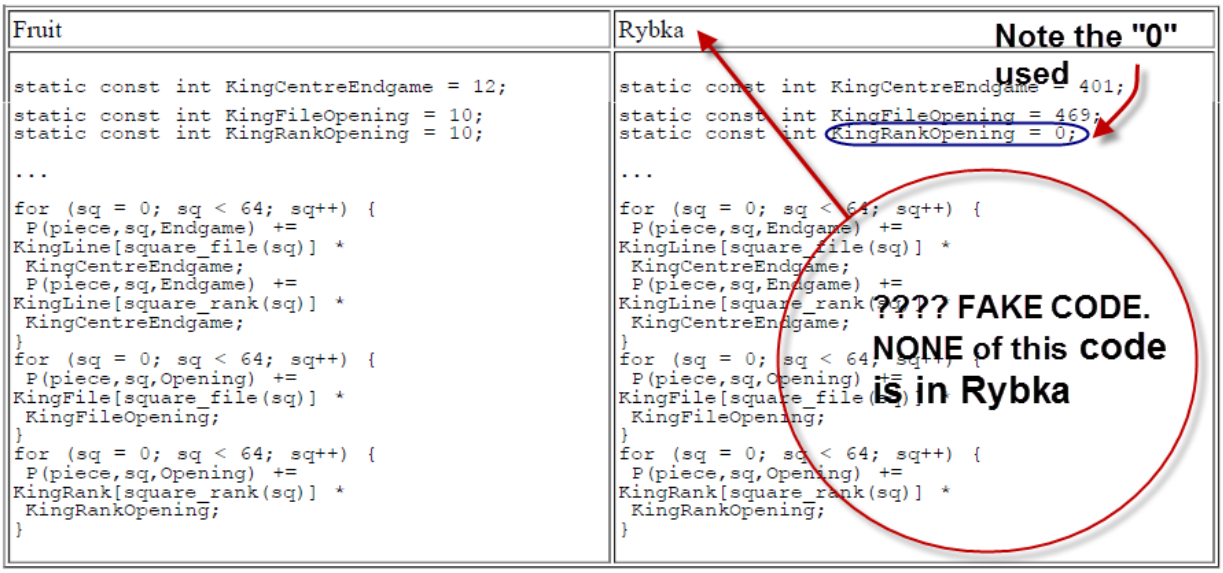
This realization led to the next insight. One of the premises of the ICGA's report is that original Rybka source code can be reconstructed from the reverse-engineered binary of Rybka 1.0 Beta. *This is simply incorrect.* The most that can be gathered from this approach is the assembly code that was output from the optimizing compiler that Rajlich used when he compiled the Rybka source code.

To restate the problem, no one has the original source code to Rybka, and Rajlich claims he only took ideas and not game-playing code from Fruit. Hyatt, et al, adamantly believe that Rajlich copied Fruit source code but cannot prove it as they do not have the actual Rybka source code. It is *absolutely not possible* to reconstruct the original source code of Rybka from a reverse-engineered executable of Rybka because there is a one-to-many mapping during this process. There are, in fact, *many* ways that Rajlich could have written Rybka and it is impossible to say exactly which path he took. This was confirmed by Dr. Hyatt himself:

> *We are not trying to take a binary executable and turn it into C. That is a one to MANY (MANY = INFINITE) mapping.*

The ICGA report shows us five pages of near-identical code, showing actual Fruit code on the left column and "Rybka" code on the right. However, as already stated, the "Rybka" code *cannot* be the original Rybka source code.

Here's an actual example of falsification in the ICGA's report:




```
static const int KingCentreEndgame = 12;
static const int KingFileOpening = 10;
static const int KingRankOpening = 10;

...

for (sq = 0; sq < 64; sq++) {
 P(piece,sq,Endgame) +=
KingLine[square_file(sq)] *
 KingCentreEndgame;
 P(piece,sq,Endgame) +=
KingLine[square_rank(sq)] *
 KingCentreEndgame;
}
for (sq = 0; sq < 64; sq++) {
 P(piece,sq,Opening) +=
KingFile[square_file(sq)] *
 KingFileOpening;
}
for (sq = 0; sq < 64; sq++) {
 P(piece,sq,Opening) +=
KingRank[square_rank(sq)] *
 KingRankOpening;
}
```

```
static const int KingCentreEndgame = 401;
static const int KingFileOpening = 469;
static const int KingRankOpening = 0;

...

for (sq = 0; sq < 64; sq++) {
 P(piece,sq,Endgame) +=
KingLine[square_file(sq)] *
 KingCentreEndgame;
 P(piece,sq,Endgame) +=
KingLine[square_rank(sq)] *
 KingCentreEndgame;
}
for (sq = 0; sq < 64; sq++) {
 P(piece,sq,Opening) +=
KingFile[square_file(sq)] *
 KingFileOpening;
}
for (sq = 0; sq < 64; sq++) {
 P(piece,sq,Opening) +=
KingRank[square_rank(sq)] *
 KingRankOpening;
}
```

As already mentioned, the first thing to note is that *none* of the code in the Rybka column is actually in Rybka. But please note the zero weight on the third line of the manufactured Rybka code. Are we supposed to believe that the weight is multiplied by zero in line three of the static declarations? If this were actually so it would not appear in the executables because any compiler would optimize and remove the unnecessary step. The only conclusion someone familiar with programming can make is that this code is fictitious. Since no one has a copy of the Rybka 1.0 Beta source code, no one can prove otherwise.

Had the ICGA titled the right-hand column of its PST analysis "functionally-equivalent code possibly used by Rybka" that would still be misleading as all that would be compared would be schematic PSTs with low information content. It would be misleading but at least there would be truth in labeling.

However, writing "Rybka" as the column title is *completely* misleading, which brings me to a crucial train of logic which seems inescapable.

It is very reasonable to conclude that the ICGA members who drafted the report knew ***exactly*** the desired effect that labeling pages of speculative material "Rybka" would have. The writers of the ICGA report are not men who act capriciously; they have long experience in academia and are intimately familiar with the standards of scientific evidence.

They could not have failed to intuit that most people lack the technical expertise and the time to comprehensively audit and assess technical documents.

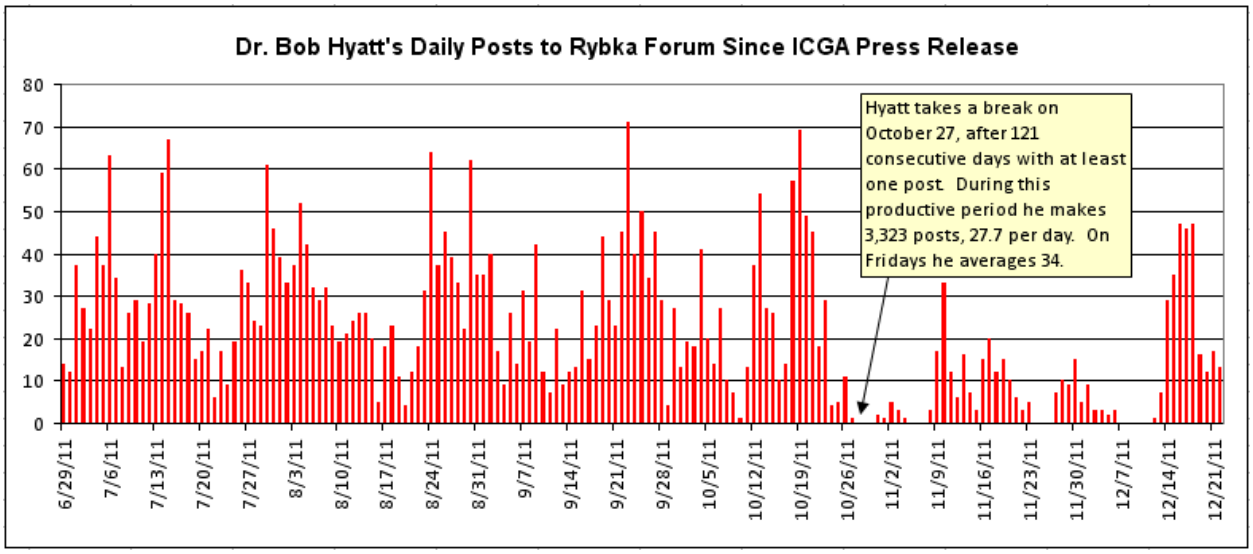
They must have known that the public at large would trust the pedigree, reputation and integrity of the report-writers as well as the ICGA as an institution.

They surely knew that very few people would have the resources or incentive to challenge a report with the ICGA's imprimatur.

Being familiar with human nature, they must have realized that those with any doubts would likely conclude that this was Rajlich's fight, not theirs, and that in any event Rajlich could provide the answers—or not.

And finally, on some level Dr. Hyatt in particular must have known that for Rajlich to fight the charges would degenerate into an unseemly quarrel where the mild-mannered Rajlich would be assailed by an unending hail of accusations, insults and sophistries.

I can say this because I and others have publicly defended Rajlich, and that is *exactly* what has happened over the course of thousands of Dr. Hyatt's posts. In my capacity as Rybka forum moderator I have access to posting statistics. The chart below speaks for itself. Four months of relentless attacks on Rajlich's own website!



These observations are not personal; they are simply factual evidence of the singular intensity and apparent motivation of Rajlich's chief accuser. Imagine how long it would take you to write 40 lucid forum posts in one day. Dr. Hyatt achieved this stupendous level of vitriol no fewer than 26 times in a four month span, peaking at 71 posts. Yet, Dr. Hyatt believes this is perfectly normal behavior for an associate professor of computer science and is not a relevant datum. I mention it because I think the reading public may have justifiable concern about Dr. Hyatt's excessive devotion to the Rajlich-is-Guilty crusade.

Dr. Hyatt's explanation

Returning to the truth-in-labeling issue, the ICGA's blanket excuse in their report is that they did note that the comparisons were not actual source code:

> *The code shown here is simply the functional equivalent; it calculates the Rybka PSTs.*

There are two problems with this. There are many different ways to write functionally equivalent code that looks nothing like the Fruit code. Dr. Miguel Ballicora posted source code to the

Rybka forum that generates Rybka and Fruit PSTs but looks completely different from Fruit code.

But more importantly, for a protracted period of time following the release of the report, Dr. Hyatt repeatedly stated that Rybka made a *direct* copy of Fruit, and referred readers to the report to prove his case, citing the side-by-side comparisons shown there as functionally equivalent to DNA evidence. He did not even concede the "functional equivalency" cited in the report until this point was brought to his attention. That is really problematic because it calls into question how the evidence was "sold" to the rest of the ICGA panel.

Over the course of the forum debates Dr. Hyatt made a series of three remarkable statements which tell us what actually happened.

Statement 1 - 26 July 2011
http://rybkaforum.net/cgi-bin/rybkaforum/topic_show.pl?pid=354882#pid354882

> *The evidence is _not_ based on "conjecture". It is based on specific analysis of Rybka and Crafty or Rybka and Fruit. There is no "interpretation" required. Have you actually _read_ Zach's and Mark's report? People keep saying "show me side by side comparisons." First page of Zach's report has _exactly_ that. Two columns. The comparison goes on for pages and pages. Side by side. Piece by piece...*

Statement 2 - 29 July 2011
http://rybkaforum.net/cgi-bin/rybkaforum/topic_show.pl?pid=355814#pid355814

> *you realize that the code on the right is imaginary? It is the code on the left, with the weights modified, so that you get the same PST values that Rybka _actually_ uses.*

Statement 3 - 7 October 2011
http://rybkaforum.net/cgi-bin/rybkaforum/topic_show.pl?pid=373948#pid373948

> *The easiest way to show a layperson that the Fruit source matches the Rybka binary is to make our "pseudo-Rybka source" match Fruit as closely as possible.*

This may be a good moment to take two aspirin pills. Let's summarize these statements:

1. There is Fruit and Rybka code side by side in the report. Pages and pages of it.
2. OK, we admit the Rybka code was "imaginary", with "weights modified".
3. OK, we now admit the Rybka code we imagined with weights modified was deliberately manipulated to look identical to Fruit code. (!)

It is clear from the time-lines that position #1 above (the side by side, piece by piece, five pages of Fruitified Rybka code) must have been the unchallenged position presented by Hyatt and Lefler to the ICGA investigating panel of 34. They must have drawn and disseminated a damning but entirely false conclusion from their own report, for how else could Hyatt and Lefler still be erroneously and misleadingly claiming there is Fruit and real Rybka code side by side in the report several weeks after publication?

Caught in this web of his own making, at one point Dr. Hyatt even admitted that the PSTs in the Rybka column were *not* copied code, boldly asserting:

> *There was NO CODE COPYING for the PST issue. NONE. NADA. ZILCH. ZIPPO.*

An emphatic statement! But two days later, apparently realizing that such a statement would mean that Rajlich was innocent, Hyatt changed his mind and wrote that Rajlich *had* copied PST code after all.

Variations in C-Sharp

According to Rajlich, he wrote a utility program (separate from Rybka and not available to users) in the C# language to generate his PSTs. As Fruit is written in C (not C#) this means there is a 100% certainty that Rajlich did not copy the Fruit PST generation code. Even if Rajlich had used similar formulas to those used in Fruit this would constitute idea re-use and not code copying. It is also quite possible that Rajlich used completely different formulas to the ones used in Fruit as demonstrated by Dr. Miguel Ballicora.

Another point is that Rajlich used his own header files (a variation on C++ templates) in the evaluation function of Rybka. These generated characteristic code repetitions in the compiler output. In the early stages of their investigation, programmer Zach Wegner mused publicly about strange repetitions in the Rybka code, but nobody understood what they signified.

It turns out Rajlich wrote large parts of Rybka using custom code (in C #defines) which allowed him to, for example, create a representation for a “good white knight” and then use exactly the same #define code to create a representations for a “good black knight”—a chess-programming cookie-cutter if you like. In so doing Rajlich effectively extended the C programming language with helpful new chess-related constructs. Rajlich commented to me on his template approach:

> *I took my "template" approach further and further over time. In Rybka 1 I was using this for things like evaluation and attack generation. Later I used it for move generation as well. Now I use it for all kinds of crazy things.*

This language extension he talks about is conceptually similar to a function key on a keyboard or a mathematical function. He is taking something complex, that may involve many steps or words, and reducing them to something much simpler. In doing this he in effect creates his own language, a lot like someone who coins a precise new word to express an idea that formerly required twenty imprecise and muddled words to describe.

The point is that Rajlich’s implementation of #defines was conceptually and functionally different from Fruit’s. While this development ethos does not necessarily constitute a defense against code-copying (his #defines could have been copy and pasted from a different source) this does represent a clear conceptual and architectural difference between the ways that Rybka and Fruit were developed.

The small window of opportunity argument

The ICGA report cites the extraordinary increase in Rybka’s strength following Fruit’s release as circumstantial evidence of plagiarism, at first glance a provocative line of reasoning. However, as has already been argued, this is a wrong diagnostic. All *modern* chess engines come into existence as fast-climbers and this strength-increase pattern cannot be credibly used as *prima facie* evidence of plagiarism.

The ICGA report presented its case as if Rajlich was only given a small time window to learn from Fruit (six months from the mid-2005 Fruit-release to the December 2005 release of Rybka

1.0 Beta), implying that he must have copied from Fruit wholesale in order to achieve the Elo strength he attained by the end of 2005, as otherwise he could never have achieved so much in six months. The report fails to mention that Rajlich became a full-time chess programmer in 2003, and earlier versions of Fruit were released starting in mid-2004. Rajlich had more than a year to study the programming style, ideas and algorithms in Fruit. It is perfectly reasonable to think that he integrated what he learned into his own program. The evidence does not justify an inference that he *must* have copied code.

The ICGA's problematic handling of the case

Apart from the substantive claims made by the ICGA a dispassionate observer ought to reflect on whether the structure and process of the investigation as well as its conclusions were reasonable and proportional to the alleged rule violation.

The ICGA decided to mount an investigation of Rybka after sixteen programmers submitted an open letter wherein they claimed Rybka contained illegal Fruit code. As I have already cited, Rajlich had himself already stated that he "went through the Fruit 2.1 source code forwards and backwards and took many things". Rajlich's statement was widely known and had been discussed *ad nauseum* by programmers and computer enthusiasts on Internet fora for a number of years. Yet, the ICGA final report pointedly omitted any mention of Rajlich's past published statements. Thus it is not incorrect to say that the ICGA was in effect investigating what Rajlich had already told the general public five years earlier, before Rybka participated in any WCCC tournaments.

A panel was formed. Dr. Hyatt served as panel gatekeeper and determined who was and was not allowed to participate. Rybka competitors, individuals with obvious conflicts of interest, and individuals who had publicly expressed their predetermined conclusion of guilt were allowed to join the investigation. The fact that such members could not only prejudice the investigation but also vote was not considered inappropriate. The ICGA defends this state of affairs by saying, in effect, 'who else but the interested parties would serve on such a panel?' This attitude, I think, is a classic example of losing the plot.

While this jury-stacking was going on the president of ICGA, Dr. David Levy, made a preemptive declaration of Rajlich's guilt in a ChessVibes article before his own panel had had sufficient time to investigate and fully deliberate the facts.

Not even half of the original committee of 34 voted for a guilty verdict. Was it even clear in advance how many guilty votes were needed to convict?

Members on the panel were only asked to decide the issue of guilt or innocence. They had no influence on the kind of penalty that would be handed down were they to find Rajlich guilty. The matter of sentencing was in the hands of the ICGA's board, headed by Dr. Levy. Levy, given his position in the ICGA, his public statement of Rajlich's guilt, and the superficially persuasive nature of the ICGA report, could hardly have been contradicted by his own board. In the end, Levy duly exercised his punitive powers based on the consensus that had been reached.

While no one questions the fact that the ICGA gave Rajlich ample opportunity to respond to their charges and he did not, there is much more to the matter than "we queried him and he did not respond." Rajlich was not merely queried. He was *publicly accused* by the head of the ICGA and *publicly excoriated* by a group of individuals who stirred themselves up into a

crusading lynch mob. A pile of "evidence" was jubilantly thrown together based on a passionately-held predetermined conclusion of code-copying which happened to be wholly at variance with actual reality. And *then* Rajlich was offered the opportunity to formally respond.

The whole process was an unprofessional disgrace.

There are those who will say, "if Rajlich had only cooperated with the ICGA investigation it would never have come to this." My response is that if you have confidence in the integrity and objectivity of the investigators this would be a compelling point. But in the absence of this confidence a perfectly reasonable attitude is "why should a four-time world champion and the world's leading chess programmer dignify the ICGA's allegations with a reply if he *knows* them with 100% certainty to be not only false, but ridiculous?" Let's put ourselves in Rajlich's shoes. Most of us, I would guess, would become belligerent and combative, and attempt to cleanse our besmirched reputation: we would strike back at our foes. That is the normal, red-blooded response of a common man. I submit that Rajlich is an uncommon man.

As for the nature of the punishment meted out by the ICGA, we might observe that justice can be defined as *every man getting his due* and *letting the punishment fit the crime*. There is no evidence that justice was done in this case in either sense, which is why I wrote this article: to publicly address an injustice and, perhaps, remedy it.

We all know that in competitive sports the players often push rules to their limits. We all know the difference between hard but clean play, yellow card offenses and red card offenses. We all know that cheating that merits a red card is deliberate, not trifling and often premeditated. Unintentional rule infractions, or even attempts to push a vague rule to its limits, do not warrant a red card, let alone something even beyond a red card: a lifetime ban. And finally, we know that rule violations, if they occur, do not merit the equivalent of capital punishment rulings five years after the fact!

A subjective view of what really went down

We finally come to a realistic appraisal of the situation in computer chess just prior to the emergence of the Rybka allegations. The demonstrated lack of proportionality in Rybka's banishment returns us again to the matter of Rybka's near-monopoly over computer chess competitions and chess engine commerce for a number of years.

Not only did Rybka have a massive lead in tournament play, but it had access to massive hardware and its latest Rybka Cluster developments were locked up, beyond the reach of reverse-engineers. Rybka's opening book was (and is) among the world's best. Leading the team was Rajlich himself, a hypercompetitive genius with an insatiable desire to win and win again, and a business model that methodically froze out everyone else. He had no friends in his peer group to watch his back because he had no peers. Moreover, he was not good at concealing that he had no use for them. It is easy to see how some could perceive this as arrogance because maybe that's exactly what it was.

It is reasonable to conclude that this dominance was so pronounced and seemed so insurmountable to Rajlich's rivals that they seized the only available opportunity to banish Rajlich and Rybka forever, not merely from ICGA-sponsored tournaments, but ***all*** tournaments anywhere in the world. If it cannot rightly be said that they actively "seized" the opportunity,

then it can more accurately be said that they passively did not regret seeing Rajlich excluded and did nothing to prevent the travesty that took place—and they voted against him.

It is also reasonable to conclude that other programmers found it unacceptable to attend week-long WCCC tournaments in far-off places like Beijing, China and Kanazawa, Japan out of their own funds, paying entry fees, air fare, hotel, food and incidentals, only to be repeatedly blown off the board by a program whose dominance seemed to increase year after year with no end in sight. The economics of this no-win proposition understandably did not work for them. This, in turn, undoubtedly threatened the near-term viability of the ICGA's annual tournament. Rybka was just in a class by itself, everyone knew it, and this apparently intractable fact simply became intolerable.

The ICGA had, in fact, already tried to constrain Rybka's superiority by limiting the amount of hardware a contestant could use in the 2009 WCCC in Pamplona, Spain. This limitation would have excluded the Rybka Cluster from competition and "leveled the playing field". Ultimately it made no difference: Rybka won the "limited hardware" tournament rather easily. Stronger measures were needed to knock Rajlich off his perch.

Finally, it is reasonable to conclude that Rajlich's long reign at the top of the rating lists, his monolithic dominance in public tournaments, sequence of menacing strategic actions such as his development of the Rybka Cluster, his publicly-stated intention to sequester his best development work so that it could not be reverse-engineered, his business alliances with Convekta and ChessBase and publicity juggernaut—all of these things and more marked Rajlich as a convention-flaunting rogue programmer and hence, in the eyes of some, a public enemy. Jonathan Swift put it cruelly:

> *When a true genius appears, you can know him by this sign: that all the dunces are in a confederacy against him.*

I concede that all the above may not be *exactly* what happened. It is only my opinion. But to this observer it is a believable narrative because it is informed by knowledge of human nature and human history. Whenever institutions or persons come along who assume a position of overwhelming power alliances of the downtrodden tend to form and start plotting the tyrant's overthrow. We see several examples of this in recent world history. Computer chess may not be a field of great sociopolitical significance but those who dwell in it have the same hard-wired human impulses. It doesn't matter that the perceived-tyrant is an innovative genius. Caesar was assassinated, after all.

In something of a surprise epilogue that took place as this article was in its final stages of being written, it emerged that the times they are a-changing for computer chess generally and the ICGA in particular. At the recent Rybka-less 2011 WCCC held in the Netherlands none of the top seven ranked programs on the then-existing CCRL 40/40 list attended, nor did the two winner-by-default co-champions from 2010's WCCC (Rondo and Thinker), omissions which stimulate furtive doubts about the credibility of the "World Championship" title the contestants struggled so mightily to win. During this competition the programmers met and some expressed a desire to change WCCC Rule 2. This was posted to the Talkchess forum by an attendee:

> *This position [100% originality] is not an obvious majority opinion anymore from the tri-ennial ICGA meeting this week where this was a lengthy agenda point.* ***A fair group of participating programmers present have expressed they want the rules to be updated.***

One line of thinking is that attribution plus added value should be sufficient to compete, instead of 100% originality.

The ICGA's policy appears to be that the programmers decide on the rules, because if there are no programmers, there can be no tournament. Without question, updating WCCC Rule 2 to reflect contemporary reality would be a years-overdue positive step. However, without justice for Rajlich as a first step any proposed rule amendment would mean Rajlich would continue in his ICGA-imposed pariah status while other programmers would be free to use Rajlich's ideas, algorithms and reverse-engineered source code (from existing and future editions of Rybka) with little fear of reprisal. First things must come first: the ICGA must retract a grave injustice inflicted upon a great chess programmer, world champion and innocent man.

Acknowledgements

Thanks to Ed Schröder for encouraging me to write this article as well as his insights on the computer chess scene going back decades. A special thanks to Nelson Hernandez, Nick Carlin, Chris Whittington, Sven Schüle and Alan Sassler for their first class editing as well as their many valuable suggestions. Without the lively collaboration of these individuals spanning several weeks this paper could not have been written. Finally, let me thank Vasik Rajlich for his clarification of various technical points and contemporaneous notes.

Thanks also to Dann Corbit, Miguel Ballicora, Rasmus Lerchedahl Petersen, Cock de Gorter, Jiri Dufek for their excellent suggestions and eagle-eyed proof reading.