32nd European Modeling & Simulation Symposium
17th International Multidisciplinary Modeling & Simulation Multiconference

2724-0029 © 2020 The Authors.
doi: xx.xxxx/xxxx

Which Trading Agent is Best? Using a Threaded Parallel Simulation of a Financial Market Changes the Pecking-Order.

Michael Rollins[1] and Dave Cliff[1,*]

[1]Department of Computer Science, The University of Bristol, Woodland Road, Bristol, BS8 1UB, UK

[*]Corresponding author. Email address: csdtc@bristol.ac.uk

Abstract

This paper presents novel results, generated from a new simulation model of a contemporary financial market, that cast serious doubt on the previously widely accepted view of the relative performance of various well-known public-domain automated-trading algorithms. Put simply, we show here that if you use a more realistic market simulator, then trading algorithms previously thought to be the best-performing are shown to be not as good as people think they are, and some algorithms previously thought to be poor performers can be seen to do surprisingly well. Automated trading is now entirely commonplace in most of the world's major financial markets: adaptive algorithmic trading systems operate largely autonomously, interacting with other traders (either other automated systems, or humans) via an electronic exchange platform. Various public-domain trading algorithms have been proposed over the past 25 years in a kind of arms-race, where each new trading algorithm was compared to the previous best, thereby establishing a "pecking order", i.e. a partially-ordered dominance hierarchy from best to worst of the various trading algorithms. Many of these algorithms were developed, tested, and evaluated using simple minimal simulations of financial markets that only very weakly approximated the fact that real markets involve many different trading systems operating asynchronously and in parallel. In this paper we take a long-established public-domain market simulator called BSE and run a set of experiments on BSE to create benchmark results from several well-known trading algorithms. BSE incorporates a very simple time-sliced approach to simulating parallelism, which has obvious known weaknesses. We then alter and extend BSE to make it *threaded*, so that different trader algorithms operate asynchronously and in parallel: we call this simulator *Threaded-BSE* (TBSE). We then re-run the trader experiments on TBSE and compare the TBSE results to our earlier benchmark results from BSE. Our comparison shows that the dominance hierarchy in our more realistic experiments is different from the one given by the original simple simulator. We conclude that simulated parallelism matters a lot, and that earlier results from simple simulations comparing different trader algorithms are no longer to be entirely trusted.

Keywords: Financial Markets; Market Simulators; Trading Agents; Simulation Methods.

1. How to use this template

This is the Word template for I3M conference paper submissions. Please note that whilst this template provides a preview of the typeset manuscript for submission, it will not necessarily be the final publication layout.

Please use the document I3M styles to prepare your manuscript.

When preparing your manuscript, be sure to change in the page headings the name of the conference you are submitting your paper to.

Use later sections starting with 'Introduction' on page 2 to write your manuscript. The remainder of this current section will provide some information for various elements you may want to include in your manuscript.

Papers that don't adhere to the guidelines provided in this template will be returned to the authors for appropriate revision. To avoid any difficulties during the typesetting process, authors must not modify any of the styles, spaces or margins.

British or American English spelling and punctuation can be used but not a mix.

1.1. Sectional headings and body text

Section headings should be left justified, bold, with the first letter capitalized and numbered consecutively, starting with the Introduction. The color of the headings is RGB(127,0,0). For your convenience, you can copy and paste one of the sample headings in this template and rename it appropriately. Please use the style named "I3M-Section-Headings" for your first-level section headings and the style "I3M-Subsection-Headings" for your second-level headings.

After the section headings, you are required to use the style named "I3M-BodyText-AfterHeading". You can use the "I3M-BodyText" style for the remaining of your paragraph. This document uses Merriweather Font and 9 points in size. The font is embedded in this document, but if you have issues with the embedded font, you can also download it from the following page https://fonts.google.com/specimen/Merriweather and install it.

1.1.1. This is a 3rd level heading

You can also use subsubsections. Please use the style named "I3M-Subsubsection-Headings" for the heading of the subsubsection.

1.2. Citations and in-text references

This is an in-text reference: (Fan and Peng, 2004) and here are two more: (Cox, 1972; Heard et al., 2006). APA in-text citations format must be used for this paper. Please consult the following website for more information: https://www.mendeley.com/guides/apa-citation-guide. You can use the style "I3M-Hyperlink" for your URLs in the paper.

Here's a citation in the text.

> This is a quote. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Please use the "I3M-Citation" style.

1.3. Symbols and Variables

Scalar variable names should normally be expressed using *italics*. Weights and measures should be expressed in SI units. All non-standard abbreviations or symbols must be defined when first mentioned, or a glossary provided.

1.4. Length of the Paper

The International Program Committee will accept pa- pers up to 8 pages (2 columns).

1.5. Bulleted and Numbered Lists

This is an example of a bulleted list.

- This is a bulleted list.
- Another point.
- A third point.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

This is an example of a bulleted list.

1. This is a numbered list.
2. Another point.
3. A third point.

1.6. Footnotes

Do not use footnotes; instead incorporate such material into the text directly or parenthetically.

1.7. Figures and Tables

All figures and tables must be numbered with Arabic numerals, have a caption and must be mentioned in the text, e.g. Figure 1 and Table 1. If you have a very wide table or figure, please see Figure 2 or Table 2. A section break should be inserted immediately before and after the figure or table and use a 1-column layout, so you'll need to carefully position it in a suitable location in your manuscript.

spaces between references. Please use "I3M-Reference" style.

2. TBSE: Threaded BSE

Much of the functionality of the Threaded Bristol Stock Exchange (TBSE) was directly borrowed from the original Bristol Stock Exchange (BSE). The key changes are contained in the way each market session operates. In BSE, each market session is executed on a single thread on which a loop executes repeatedly until the session time is exceeded. Within this loop a random trader is selected, their getorder() function executed, then if an order is generated this is processed by the exchange and each trader is given an opportunity to update their internal values with the new information.

The problem with this style of simulation is that each trader is allotted as much time as it needs to execute its getorder() function, and it is guaranteed that nothing on the Limit Order Book (LOB) will change as it executes this function. This means that the execution time of each trader has no impact on its performance, simply its ability to generate an order price that will be accepted by another trader, whilst generating the greatest profit. This differs from a real financial exchange where all traders operating on the market will be operating asynchronously. In an asynchronous market, a trader may look at the LOB, which is the currently available information on that market, and use this to calculate what it considers an optimum order price to send to the exchange. However, if this trader uses a complex, slow running algorithm to do so, it may find that by the time it has completed its calculation, another simpler and faster trader has already succeeded in executing a trade, which as a result changes the position of the market which may render the order than the complex trader has posted less profitable than expected.

The solution to this problem is to create an asynchronous exchange simulator, this is what TBSE means to do. In TBSE each trader executes on its own computational thread, as does the exchange. There is also the main thread of the Python code which continues to run during the market session and is responsible for the distribution of customer orders to each of the traders. Each trader thread consists of a loop which executes continuously until the end of the trading session. Within this loop the trader first receives information of trades which have been successfully executed by the exchange, it then updates its own internal record keeping if any of these trades involved itself and executes its respond() function to update its internal variables based on the new LOB. Once it is updated with the latest version of the LOB it then executes its getorder() function which determines whether it should post a new order to the exchange, and if so at what price. This order is then put on a queue to be sent to the exchange.

The exchange operates on the opposite of this queue, reading orders from the queue and then processing them by either adding them to the LOB or executing a trade if the order price "crosses the spread", meaning for a buy order the order price is higher than the current best ask price or vice versa for a sell order. If a trade can be executed it then places the details of this order onto a queue for each trader to read before submitting its next order. The system for processing orders and trades is almost identical to that of BSE.

2.1. Python Multi-Threading

Multi-threading is where a processor can operate multiple threads of execution concurrently. This differs from multi-processing, which Python also supports, where different processes execute simultaneously. Both are forms of parallelism, but where in multi-processor programming multiple processes are executing at the same time on different processors, only one thread can execute and any one time. The execution of each thread is divided into short segments and the processor regularly switches between each thread for a short period of time, allowing the appearance that all threads are executing at the same time. It is not guaranteed that each thread will be given the exact same amount of execution time during each cycle, but over the course of an entire program, which may involve millions of cycles, the execution time given to each thread should tend towards the same value. In TBSE this means that each trader is given the same amount of processing time, so a faster algorithm can complete its execution and start processing a second order while a slower algorithm is still calculating its first order price. In Python, the execution of threads is synchronised by a Global Interpreter Lock (GIL) which ensures that only one thread is executing at a time. Multi-threading was chosen over multi-processing as most general-purpose CPUs have at 8 processing cores, which as 40 traders are needed for each experiment on TBSE does not give a significant benefit, and adds increased complication to the development process.

The queues used within TBSE are synchronous first in, first out (FIFO) queues, meaning that the trader who places an order on the exchange queue first will have their order processed first. This is a good simulation of real exchange behaviour as it is standard for exchanges to prioritise orders by their time of arrival.

\subsection{Supply and Demand Schedules}

In previous work exploring trading agents, such as Das \& Tesauro '01, it has been common practice to draw limit prices from a fixed uniform random distribution. In Vytelingum '06, for simple market experiments, experiments were run using limit prices produced by four separate static supply and demand schedules (SDS), this method was later reused in Snashall \& Cliff '19. This allowed research into how different trading algorithms responded to different SDS. In the experiments ran for this paper, each experiment was made up of 5 random SDS each being used to generate limit prices for 100 trading periods. This was done as it is impossible to know the true underlying supply and demand curves for a market, so if a trading algorithm is optimised for a particular SDS, the results of that experiment may not reflect the way the traders would under most circumstances. Each SDS is generated from a random minimum value between 0 and 100 and a random maximum values between 100 and 200, this ensures that the resulting supply curve is ascending, and the demand curve is descending, as is standard in financial markets. Limit prices from these schedules were selected at fixed intervals between the maximum and minimum prices.

3. Results

Table 1 shows a high-level summary of our results for AAvsZIC, AAvsZIP, GDXvsZIC, and GDXvsZIP. Characterizing each of these four experiments as Algorithm A vs Algorithm B, the central subtable of Table 1 shows the count of "wins" for A and the count of wins for B in BSE, with the higher of the two counts highlighted in bold font; and then the right-hand subtable of Table 2 shows the corresponding win-counts for A and for B operating instead in TBSE, again with the higher value highlighted in bold font.

As can be seen from Table 1, our results for BSE are consistent with those previously published in the literature: there is nothing in our BSE results to challenge the status quo: AA beats both ZIC and ZIP, as does GDX.

However, the TBSE results in Table 1 tell a very different story. AA still beats ZIC, which is intuitively what one would expect. But the other three dominance relationships have been inverted: in TBSE, ZIC beats GDX while ZIP beats both AA and GDX – scoring more than twice as many wins as GDX, beating it by a larger margin than it beat ZIP in BSE.

If we chose, we could stop here. We have now established an answer to the question that we set out to explore: whether a switch to a more realistic market simulation makes a difference to the dominance relationships previously reported. This is a significant finding, and is the primary contribution of this paper.

Table 1. Summary of all our experiment results, showing total number of "wins" summed over 19 different A-vs-B trials in BSE (central sub-table) and in TBSE (right-hand sub-table). The 19 different trials vary the ratio R_a of trading algorithm A ("AlgoA") to trading algorithm B ("AlgoB") from 1:19 through 10:10 to 19:1, and at each ratio we conduct n=500 i.i.d market sessions, which are treated as contest between AlgoA and AlgoB: if, at the end of a session, the average profit per trader for AlgoA is greater than that for AlgoB, then that counts as a "win" for AlgoA. Hence, the maximum possible score is 19x500=9,500. In each row, bold-font text is used to highlight the larger number of wins in each A-vs-B comparison. As can be seen, switching from BSE to TBSE has no effect in the case of AA-vs-ZIC, but for the other three cases we see a reversal of the dominance relationship. See text for further discussion.

AlgoA	AlgoB	BSE # A Wins	BSE # B Wins	TBSE # A Wins	TBSE # B Wins
AA	ZIC	**7095**	2405	**7370**	2130
AA	ZIP	**7581**	1739	4383	**5117**
GDX	ZIC	**5517**	3988	4300	**5199**
GDX	ZIP	**6538**	2962	2574	**6926**

The rest of this results section shows selected highlights from digging deeper into the results in our experiment. As it happens, a deeper dig unearths some thought-provoking results.

Recall that each number in Table 1 represents the number of 'wins" scored by a specific algorithm, in a series of 19x500 market sessions, ranging over 19 values of R_a (from 1:19 to 19:1), with n=500 trials at each value of R_a -- one obvious thing to do is to separate the aggregate results for any one pair of algorithms into the 19 sets of results, one for each value of R_a, and to look for any interesting changes in the patterns of wins as R_a is swept from one extreme to another. Table 2 shows such a data-set, for AA-vs-ZIC. As you can see, at the bottom of the table is the sum of all the wins in each column, and the relevant column-sums in Table 2 are the values that populate the first row of Table 1. Table 2 adds a third column to the results for BSE and TBSE: a variable we call *Δwins*, which is simply the difference between the two algorithm's win-counts. In an A-vs-B comparison of two trading algorithms, if *Δwins*>0 then A outperforms B, and if *Δwins* <0 then B outperforms A. The two sets of *Δwins* values in Table 2 are plotted graphically in Figure 1, a paired plot that we refer to as the *delta curves* for two trading strategies tested in BSE and TBSE.

Table 2. The data that was aggregated into the top row of Table 1, tabulated to show the individual results from each of the 19 different ratios of the two algorithms used: n=500 at each ratio. Column sums are displayed at the bottom of the table, and correspond to the values given in the top row of Table 1.

	BSE			TBSE		
Ratio	AA Wins	ZIC Wins	Δ	AA Wins	ZIC Wins	Δ
1:19	**279**	221	58	**297**	203	94
2:18	**355**	145	210	**357**	143	214
3:17	**352**	148	204	**375**	125	250
4:16	**377**	123	254	**384**	116	268
5:15	**376**	124	252	**377**	123	254
6:14	**408**	92	316	**415**	85	330
7:13	**362**	138	224	**410**	90	320
8:12	**401**	99	302	**417**	83	334
9:11	**370**	130	240	**407**	93	314
10:10	**395**	105	290	**414**	86	328
11:9	**408**	92	316	**388**	112	276
12:8	**438**	62	376	**419**	81	338
13:7	**346**	154	192	**398**	102	296
14:6	**395**	105	290	**421**	79	342
15:5	**420**	80	340	**377**	123	254
16:4	**355**	145	210	**368**	132	236
17:3	**373**	127	246	**388**	112	276
18:2	**361**	139	222	**412**	88	324
19:1	**324**	176	148	**346**	154	192
Sum	**7095**	2405	4690	**7370**	2130	5240

The pair of delta curves in Figure 1 look to be approximately the same, modulo some noise (which is to be expected given the stochastic nature of our experiments). The fact that there is no great difference is no surprise, given that in both BSE and TBSE our results show AA beating ZIC.

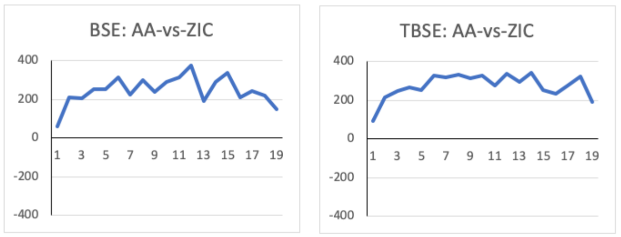

Figure 1. "Delta Curves" for AA and ZIC in BSE (left) and TBSE (right). This pair of graphs shows the *Δwins* data tabulated in Table 2: the horizontal axis shows R_a and is labelled with the number of AA traders in the ratio; the vertical axis is the value of *Δwins.* When *Δwins>0*, AA outperforms ZIC, and when *Δwins<0,* ZIC beats AA.

Figures 2, 3, and 4 show the delta curves for AA-vs-ZIP, GDX-vs-ZIC, and GDX-vs-ZIP, respectively. These three figures are each derived from the relevant data in the same way that Figure 1 was derived from the data in Table 2. Because these are the three sets of experiments in which the switch from BSE to TBSE inverted the dominance relationship between the two trading algorithms, the delta curves deserve some examination and discussion.

AA-vis-ZIP (Figure 2). Here there seems to be some coherent structure in the BSE delta curve: although AA consistently outperforms ZIP, the degree by which it beats ZIP seems to attenuate when the R_a is at either extreme of its range, and the maximum outperformance of AA over ZIP seems to be when R_a<10:10, i.e. when AA is in the minority of the population. The relevance of this is seen in the TBSE delta curve in Figure2: we saw in Table 1 that ZIP wins on aggregate in this set of experiments, but Figure 2 makes clear that actually when AA is in a small minority in TBSE it can still outperform ZIP, but as soon as the proportion of AA traders starts to approach 50% its dominance disappears and ZIP consistently outperforms AA as soon as AA is in the majority.

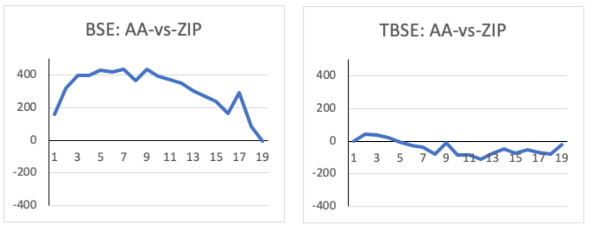

Figure 2. Delta curves for AA vs ZIP; format is the same as in Figure 1.

GDX-vis-ZIC (Figure 3). Here, for both BSE and TBSE, the delta curves have a clear coherent structure to them, which is something that we do not think has been reported before in the literature: we know from Vach (year), Cliff (2019), and Snashall & Cliff (2019) that ratio matters, but none of those publications reported or explored such strongly coherent relationships between ratio and results, between R_a and $\Delta wins$. Figure 3 shows fairly unambiguously that GDX outperforms ZIC when GDX is in the minority, and ZIC outperforms GDX when ZIC is in the minority. Figure 3 also reveals that the qualitative nature of the swing from GDX dominance in BSE to ZIC dominance in TBSE differs from that from AA to ZIP that was illustrated in Figure 2: whereas the two delta plots in Figure 2 are markedly different, the two curves in Figure 3 are remarkably similar: the TBSE curve could plausibly be described as what happens when the BSE curve is shifted slightly to the right and slightly down. Also notable is that in both BSE and TBSE the maximum outperformance of ZIC by GDX happens at roughly a ratio of 1:3, and the maximum outperformance of GDX by ZICs seems similarly to happen at roughly 3:1. This strikes us as curious in that experience tells us that usually in these kind of experiments maxima would normally be expected to occur either at the endpoints of the scale (i.e. at ratios of 1:19 or 19:1) or near the midpoint (i.e. at 10:10) why this is so is something we aim to investigate in further work. *[To be reported in the 10-page version of this paper]*

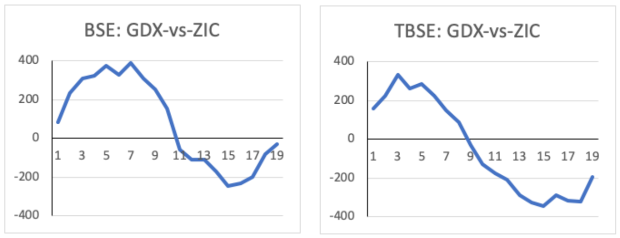

Figure 3. Delta curves for GDX vs ZIC; format is the same as in Figure 1.

GDX-vis-ZIP (Figure 4): again there is a clear coherence to the delta curves, although the relationship between the BSE and TBSE curves is not as similar as that in Figure 3, and not as different as that in Figure 2. In a point of similarity with the GDX-vs-ZIC curves, again the peak performance of the two strategies come at ratios of roughly 3:1 and 1:3 (and, again, we do not know why this should be so) but for GDx-vs-ZIP the relative performance of the two algorithms does not level out to zero near ratio values of 10:10 and then cross into underperformance for the algorithm that is in the majority; instead in BSE GDX pretty consistently outperforms ZIP (albeit with the degree of outperformance steadily falling as the proportion of GDX traders increases, and then crossing over into underperformance at ratios around 3:1) and in TBSE the situation is the inverse: now ZIP is the dominant algorithm at almost all ratios, and again the degree of dominance falls steadily as the proportion of ZIPs increases beyond 1:3). Once again, further work will be required to understand why these delta curves have this particular qualitative shape.

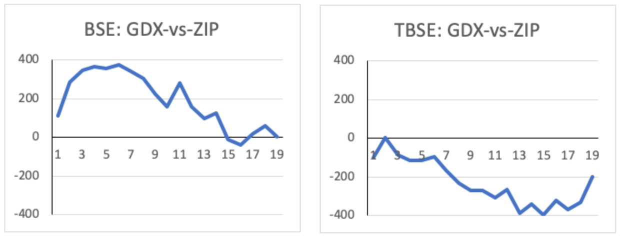

Figure 4. Delta curves for GDX vs ZIP; format is the same as in Figure 1.

[Note: in the 10-page version of this paper we will present additional results, in the same style as shown here, illustrating the nature of the interactions between other pairs of trading strategies not shown here, such as AA-GDX and ZIC-ZIP]

4. Discussion and Further Work

One thing that is notable from the delta curves in Figures 2, 3, and 4 is that – despite the points of similarity highlighted in the discussion – the effect that shifting from BSE to TBSE has on the delta curves is qualitatively different for each pair of trading algorithms: the data we have studied thus far reveals no conveniently simple patterns that allow us to make a priori predictions or generalizations about when Algorithm A will outperform Algorithm B. Figure 3 is the starkest illustration of this point: recall that GDX was developed at IBM TJ Watson Research Labs, was preceded by two earlier versions (GD, then MGD), and when published was described by IBM as "..the best-performing trading algorithm in the currently published literature." We do not seek to criticize the IBM team, but our results show that, when the conditions are right (i.e., when the ratio of GDX:ZIC is in the right range), in fact GDX can be consistently out-performed by ZIC. And Gode & Sunder's ZIC paper had been published six years before IBM's GDX paper, and that whereas GDX mixes the construction of a probabilistic belief function with techniques from dynamic programming, the ZIC algorithm is so extremely simple that it can be written in only one line of program-code.

One obvious avenue of further work is to attempt to understand what features, if any, of the trading algorithms interact in such a way that they give rise to the delta curves that we have plotted here, and the extent to which those delta curves are affected by changes in other significant factors, such as the market's supply and demand functions, or the total number of traders in the market. This paper has presented results that focus on four trading algorithms that have been used repeatedly in studies of artificial trading systems over the past 20 years, but there are several other algorithms which could be added to this analysis, including Gjerstad's (2003) HBL; the Roth-Erev algorithm (REF) such as the simple GVWY and SHVR built into BSE; and the ZIP-related ASAD (REF). And, when we're done with analyzing interactions between pairs, between two types trading algorithms, we can move on to triples (in which case the delta curves could be plotted as points on a simplex), and then on to various ratios of four or five or six different trading algorithms, and so on. But as the number of algorithms involved in any one comparison increases, so do the number of trials required (the combinatorics are explosive, and the computational cost even of the experiments shown here was measured in days of CPU time), and so do the difficulties of visualizing and comprehending the results.

5. Conclusion

Prior to the publication of this paper, anyone reading the trading-agent literature would have understood that there was general agreement that, in general, AA beats GDX, GDX beats ZIP, and ZIP beats ZIC. Only a careful reading of the literature would reveal that most of these results came from single-threaded simulations. Our results presented here show that while the AA>GDX>ZIP>ZIC dominance hierarchy may hold true in simple simulations, as soon as real-time factors matter, as soon as the various algorithms are operating in parallel, the computational costs of a sophisticated algorithm such as GDX count against it, and under the right conditions it can be outmanoeuvred by a simpler but faster algorithm. So, the primary contribution of this paper is our demonstration here that the old single-threaded dominance hierarchy is not maintained in multi-threaded TBSE. As Table 1 shows, in TBSE we have AA beating ZIC, ZIP beating AA, ZIC beating GDX, and ZIP beating GDX; a set of results that could be summarized as ZIP>AA>ZIC>GDX.

While it would certainly satisfy our intellectual curiosity to explore various further aspects of this work in more detail, we are painfully aware that the final payoff for any such exploration may be nothing more than a shoulder-shrug and the concluding observation that "it's just really complicated". Real-world markets are complex systems. In fact, technically, they are complex *systems-of-systems* (see e.g. Sommerville *et al.,* YEAR): they're made up of multiple independently designed & autonomously operated *constituents* (rather than *components*); those constituents operate asynchronously and in parallel; the number of constituents changes over time as old ones leave and new ones are added; and of course many or perhaps all of the constituents are each individually nonlinear systems, interacting with other constituents in nonlinear ways. Trying to capture all of this in a simulation is challenging, but we believe that the results presented here demonstrate that the extension from BSE to TBSE, the move to parallel multithreaded asynchronous operation, is clearly a step in the right direction. And, in taking that step, we have called into question the previous results that proposed one particular trading algorithm as "the best-performing" or "the most dominant".

In fact, our work calls into question not just the truth of such statements, but also whether it is really ever worth trying to make such statements at all, because trading-algorithm performance, and hence dominance, is clearly so heavily affected by factors exogenous to the trading algorithm, chief of which is what other algorithms it is competing against, and in what proportion or ratio those different algorithms are present in the market: it's manifestly game-theoretic. Put simply, we question whether, in markets that are sufficiently realistic to be relevant to the real world, there can ever really be a single specific trading algorithm that is "dominant", that actually beats all the rest. That is: if you think your trading algorithm really is the best-performing one, we'd say that you've probably tested it in simulations that are too simple.

References

Cox, D. R. (1972). Regression models and life tables (with Discussion). *J. R. Statist. Soc.* B, 34:187–220.

Fan, J. and Peng, H. (2004). Nonconcave penalized likelihood with a diverging number of parameters. *Ann. Statist.*, 32:928–61.

Cliff ZIP 1997

Cliff BSE EMSS 2018

Cliff "modern methods" EMSS2019

Cliff ICAART 2019

Gjerstad HBL

Gode & Sunder ZIC

IBM papers

Miles & Cliff on DBSE

Roth/Erev

Snashall MEng thesis

Snashall & Cliff LNAI chapter

Vach MSc thesis

ASAD paper

Sommerville et al CACM on LSCITS

Heard, N. A., Holmes, C. C., and Stephens, D. A. (2006). A quantitative study of gene regulation involved in the immune response of Anopheline mosquitoes: An application of Bayesian hierarchical clustering of curves. *J. Am. Statist. Assoc.*, 101:18–29.