Utilising VIPER for Parameter Space Exploration in Agent Based Wealth Distribution Models

O.J.Scholten, R.J.Spick, K.A.Hawick

The School of Engineering and Computer Science, University of Hull, Cottingham Road, Hull, Yorkshire HU6 7RX

UNIVERSITY OF HULL

the DIGITAL centre

Objectives

How can high performance computing capability be harnessed to search large parameter spaces in finite time?

- Create a framework for execution, analysis, and visualization of models.
- Exploit parallelism as a means of running multiple models simultaneously.
- Evaluate existing libraries and frameworks for parallelism across compute nodes.
- Present early findings and visualizations.

Introduction

VIPER presents a novel opportunity to exploit parallelism as a means of making scientific discoveries in previously unreachable areas. One such area is agent based modelling, specifically exploring parameter spaces in wealth distribution models.

By using relevant parallelization libraries we have been able to run multiple instances of Schelling's model of segregation[4] with varying conditions and added mechanics. Outputs have then been visualized giving key insight into the model's emergent properties and the relationships between their parameters. Figure 1 shows an example of a single model run, with clusters appearing as shaded regions. All presented figures are derived from experiments run on a 500x500 grid of agents over 1x10^9 time steps which have been normalized.

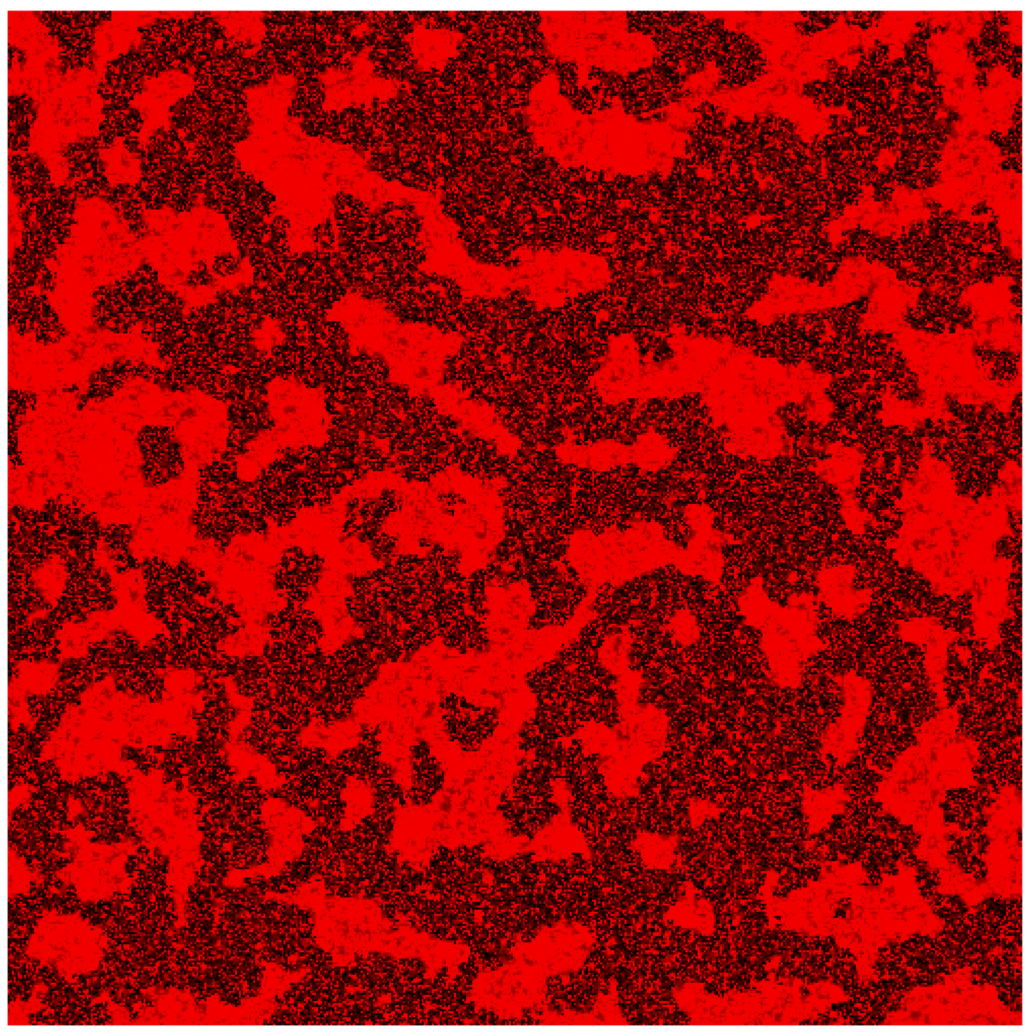

Figure 1: Segregation clusters with Schelling satisfaction of 0.29 with fixed trading fraction, at t=1x10^9.

Early Methodologies

As preparation for deployment to VIPER, multiple test versions of bespoke agent based modelling frameworks have been developed initially for use on local machines. These vary greatly in terms of their overall architectures and the languages/libraries they incorporate. Earliest work included a wrapper style program (C++) where each model would be called as a standalone program by another 'launcher' script. Whilst this structure worked in a local machine context, it could not easily be scaled for deployment on VIPER given its architecture. Scalable programs were then consolidated into a single unified framework, with parameters set in the code as opposed to through arguments.

With successfully parallelized models running efforts were made to make the process of experimentation more concise. This meant streamlining our workflow and exploring extra software and techniques to reduce the time taken from inception of a new model idea to successful execution and analysis. Through implementation of cloud based storage, collaborative code editing, and a number of other measures we have been able to condense the preliminary exploration process down to a matter of minutes as opposed to days.

VIPER Utilisation

Early VIPER usage was limited to single and multiple processors on a single node using the OpenMP library for interprocess coordination. As exploration requirements increased, OpenMPI implementations were explored allowing increased usage as high as 100 processors over multiple nodes. This work presents potential to again increase usage by a similar factor, facilitating more rigorous statistical analysis as subsequent outputs are collated.

Usage over the evolution of this project translates to approximately 50 core hours across multiple users with each run taking approximately 10 minutes of real time to execute. Figures 1 through 4 represent the results of over 40 job submissions, each more computationally intensive than the last.

Discoveries

Figure 2 shows how clustering of similar agents increases over time depending on the satisfaction value used, each line represents a single run on VIPER with the full set generated in parallel. It also clearly visualises the relative instability of the model's equilibrium state at satisfaction values lower than ∼0.42 when compared to the smoother lines at values greater than ∼0.42. The clustering coefficient

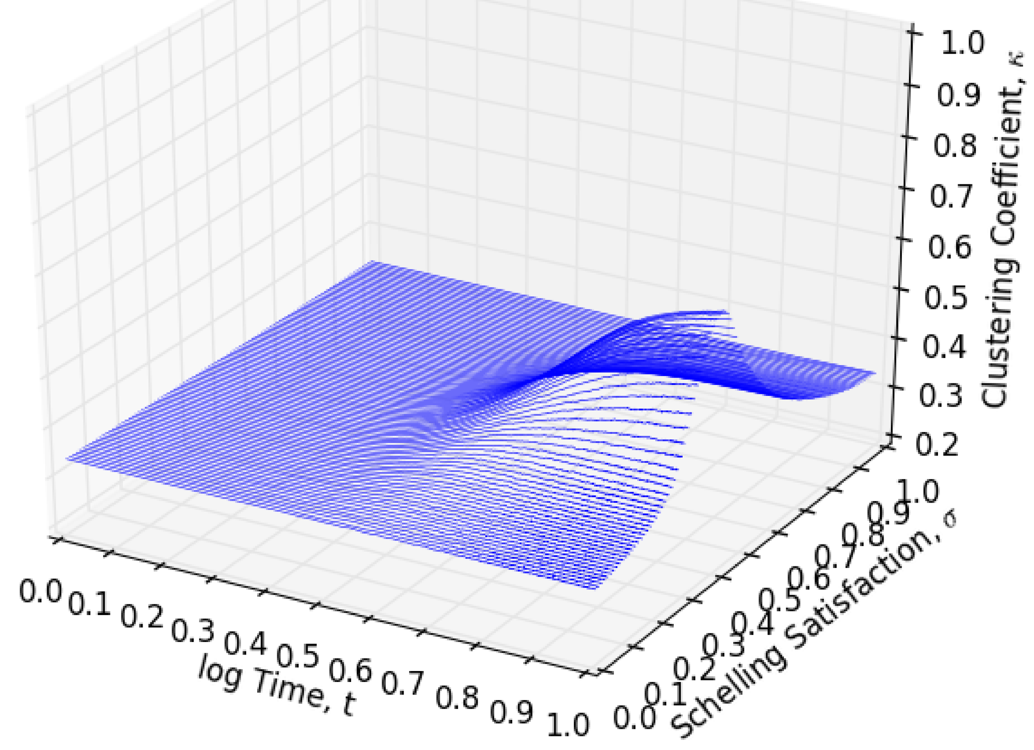

Figure 2: Clustering over time in Schelling's model with varying satisfaction values.

By adding a trading mechanic to this system the clustering coefficient at equilibrium in the majority of models is increased, in the extreme to the point of near total segregation. It is also clear that this added mechanic introduces a great deal of instability to the system's clustering, with values fluctuating by as much as 60% between measurements.

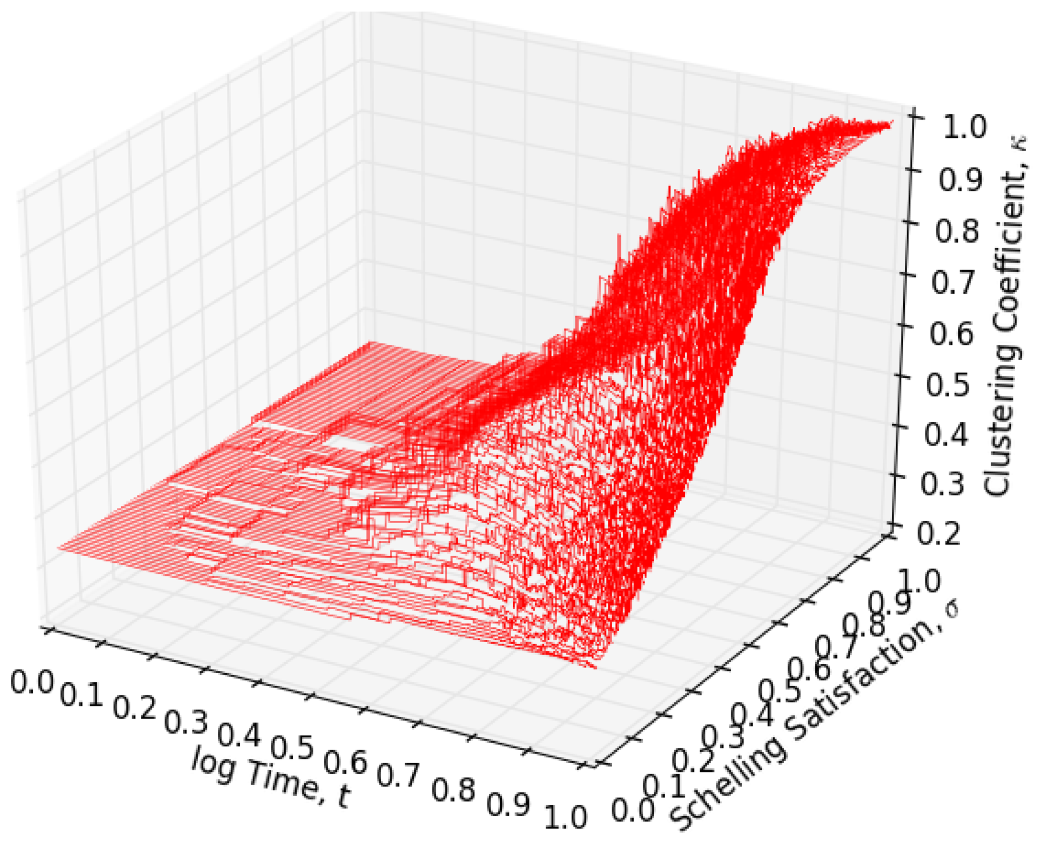

Figure 3: The effects of adding a trading mechanism to the model shown in figure 2.

With the above results considered, there is scope for the application of relevant statistical mechanics principles to quantify the effects presented by the visualization(s). This observation will form the basis of collaboration between physics and computer science researchers in further exploration. Increasing VIPER utilisation with the goal of adding error bars, averages, etc. to such noisy results will also play a key role in future work.

Visualization & Analysis

Model outputs are stored as CSV files, saved directly to the VIPER file system. These files are then transferred to a local machine for visualization and analysis using several Python scripts. As exploration efforts increase so will the need for robust and efficient visualization methods, making transfer of the data visualization computation to the VIPER nodes an important future milestone for this project.

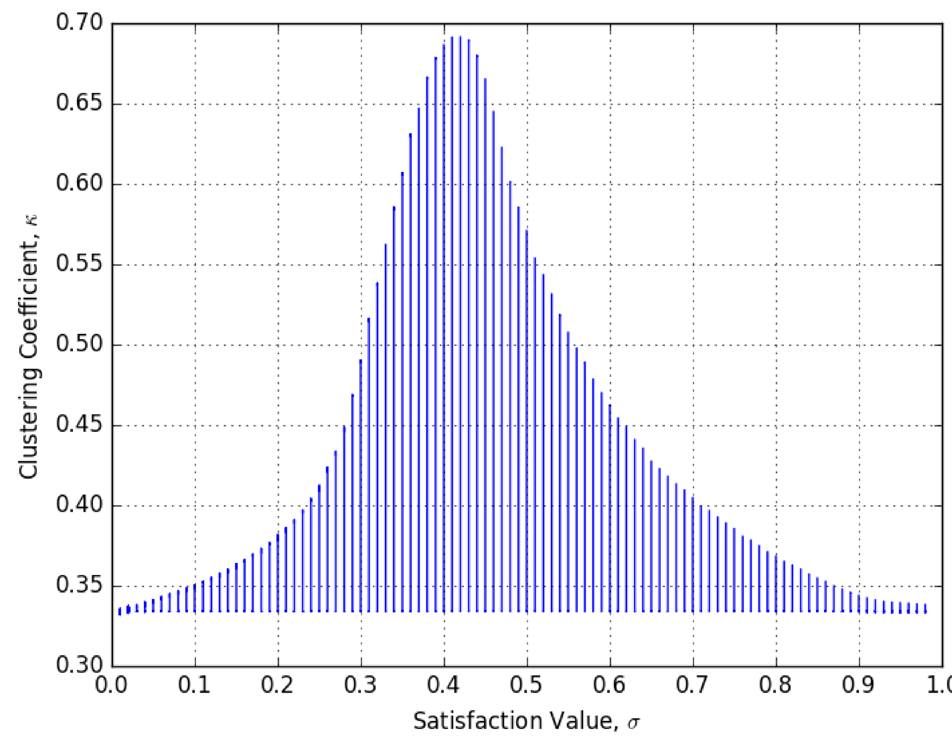

Figure 4: Cross section of figure 2 at t=1x10^9. Skew = 1.03, Kurtosis = -0.23.

Figure 4 presents an initial method of analysing the data, such 'slices' of the surface plot style data in figures 2 and 3 offer the ability to extract measurements which can then be plotted to give a much higher level view of the findings of our experiments. These include typical statistics measures such as skew and kurtosis and can be built upon over time.

Conclusion

Through utilising VIPER we have found a rich and interesting area for further work, with benefits beyond this project alone as the code base can be repurposed for other applications. The developed framework could also be shared between researchers both in the same groups and across departments within the university, promoting inter-department collaboration and increased utilisation of the high performance facilities.

Future work will include implementing more sophisticated analytical and visualization capabilities with an emphasis on increasing utilisation of VIPER as models increase in computational demand.

References

[1] Dragulescu, A. and Yakovenko, V.M., 2000. Statistical mechanics of money. The European Physical Journal B-Condensed Matter and Complex Systems, 17(4), pp.723-729.

[2] Goswami, S. and Sen, P., 2014. Agent based models for wealth distribution with preference in interaction. Physica A: Statistical Mechanics and its Applications, 415, pp.514-524.

[3] Clark, W.A., 1991. Residential preferences and neighborhood racial segregation: A test of the Schelling segregation model. Demography, 28(1), pp.1-19.

[4] Schelling TC. Dynamic models of segregation. Journal of mathematical sociology. 1971 Jul 1;1(2):143-86.

[5] Raicu, I., Foster, I.T., Zhao, Y.: Many-task computing for grids and supercomputers. In: Proc Workshop on Many-Task Computing on Grids and Supercomputers. Austin, TX, USA (17 November 2008)

[6] Inoue, J.I., Ghosh, A., Chatterjee, A. and Chakrabarti, B.K., 2015. Measuring social inequality with quantitative methodology: analytical estimates and empirical data analysis by gini and k indices. Physica A: Statistical Mechanics and its Applications, 429, pp.184-204.

[7] Wang, K., Zhou, X., Chen, H., lang, M., Raicu, I.: Next generation job management systems for extreme-scale ensemble computing. In: Proc 23rd Int Symposium on High-performance, parallel and distributed computing (HPDC). pp. 111–114. Vancouver, BC, Canada (23-27 June 2014)

[8] Liu, J., Pacitti, E., Valduriez, P., Mattoso, M.: A survey of data-intensive scientific workflow management. J. Grid Computing 13, 457–493 (2015)