International Journal of Business and Economics Research
2013; 2(3): 33-40
Published online June 10, 2013 (http:// www.sciencepublishinggroup.com/j/ijber)
doi: 10.11648/j.ijber.20130203.11

Large-scale agent-based models in marketing research: the quest for the mythical free lunch

Alexandru Voicu[1, *], **Cristina Galalae**[2, *]

[1]Faculty of Management, Bucharest Academy of Economic Studies, Bucharest, Romania
[2]Faculty of Economics and International Business, Bucharest Academy of Economic Studies, Bucharest, Romania

Email address:
alex@beyond3d.com(A. Voicu), cristina.galalae@fulbrightmail.org(C. Galalae)

To cite this article:
Alexandru Voicu, Cristina Galalae. Large-Scale Agent-Based Models in Marketing Research: The Quest for the Mythical Free Lunch. *International Journal of Business and Economics Research*. Vol. 2, No. 3, 2013, pp. 33-40. doi: 10.11648/j.ijber.20130203.11

Abstract: Positioned as an alternative to equation-based methods, agent-based modelling (ABM) has shown notable promise in dealing with cases where the latter has proven inadequate. One of the areas where the limitations of traditional approaches are most pronounced is that of consumer behaviour research. A primary trait encountered within this scope is that of adaptability, and agent-based methods appear to be ideally suited to the task of capturing this dimension. It therefore follows that marketing researchers are likely to gain novel and extensive insight by way of constructing large-scale, complex ABMs. However, the computational cost of complex simulations can be prohibitive. Furthermore, the literature makes little effort to elucidate means of making such ABMs feasible, beyond relying on natural hardware evolution. Unfortunately, the latter source of growth has grown stagnant, and the only avenue for the continued expansion of performance appears to be the move to parallel platforms and programming. The present research presents a cross-section of the current state-of-the-art in high-performance ABM frameworks, and proposes a novel approach to levering the as of yet untapped potential of cheap, ubiquitous Graphics Processing Units (GPUs). This insight is mapped into the space of consumer behaviour research, and a consistent argument is made in favour of larger, more detailed, ABMs, both as alternatives to current approaches as well as a development of prior forays into this area. In conclusion, a call to action is formulated, both to marketing researchers as well as computational economists, emphasizing the interdisciplinary requirements of ABM usage in the field of marketing.

Keywords: Agent-Based Modelling, Consumer Behaviour, Parallel Programming, GPGPU, C++ AMP, Economics

1. Introduction

One of the many challenging tasks faced by the yet young science of marketing is that of modelling and, ultimately, comprehending consumer behaviour. Indeed, it could be argued that this is actually one of the most difficult endeavours in economics research in general. This observation stems from the fact that the mathematical apparatus that economists in general, and marketers in particular, have come to rely on is woefully inadequate when it comes to representing the dynamics of complex adaptive systems[1]. Whilst it is commonly accepted that systems made up from humans fully qualify for being included in this category, few if any comprehensive tools exist that allow one to gain notable insight into their inner workings. In the complicated balancing act between tractability and informational entropy brought about by a particular modelling exercise, the casualty ends up being the latter aspect. Through the enforcement of overly restrictive assumptions (e.g. hyper-rationality, omniscience[2]) or very high-level abstract concepts (e.g. broad scope latent variables in Structural Equation Models (SEM)[3], approximation as a hydraulic system in System Dynamics[4]), models become solvable, albeit not necessarily very realistic. Agent-Based Modelling (ABM) proposes an alternative, generative approach, which focuses on the micro-specification of purposive agents (e.g. consumers, firms, managers – roles are not mutually exclusive), which through their free, uncontrolled interaction generate the dynamics of interest. Members of the scientific community seems increasingly keen on pointing out that ABM has the potential to greatly expand the scope of economics modelling, whilst at the same time increasing its depth and thus removing age-old limitations[5],[6],[7],[8].

A primary bound on the evolution of ABM in economics is due to high computational costs. Indeed, part of the surge in interest for ABM is attributed to the exponential growth in processor performance associated with the steadfast progress of manufacturing technologies, as conjectured in Moore's Law[9]. More often than not, authors tend to brush the topic of performance aside, and express the conviction that the performance scaling will continue its natural course, thus making increasingly complicated ABMs tractable[10], [11], [12]. This view is mirrored in the frameworks commonly employed by researchers: NetLogo[13], MASON[14] or Repast[15]. Whilst holding the lion's share of applications encountered in the literature, they are all geared towards usability, at the expense of performance, being written in higher level, interpreted languages (i.e. Java, or, more recently, C#), with no aggressive tuning in place. A common moniker for the sustained and constant growth of single-processor performance has been that of a "free lunch". This captures the basic idea that programs could be written sub-optimally, with hardware evolution picking up the slack and ensuring that they would eventually achieve sufficient levels of performance. Unfortunately, whilst relying on the "free lunch" was adequate up until early in the 21st century, as outlined in the first chapter of[16], power and thermal limits have all but stopped single-processor scaling. In an aptly named essay[17], Herb Sutter deals with this topic in a comprehensive manner, and indicates concurrency / parallelism as the new performance vein to be mined. Reference[18], due to the same author, fully fleshes out the sources of parallel throughput, and the various forms in which the latter can be leveraged, with emphasis being placed on heterogeneous solutions as being endowed with the most potential. Very recent research[19],[20] match this description, in recent years Graphics Processing Units (GPU) have emerged as cheap, ubiquitous solutions available to researchers from all fields. In this context, it is not unreasonable to look to GPUs and, by extension, the General Programming of GPUs (GPGPU) paradigm as future avenues for supporting ABM development. Intriguingly, the literature on the topic is relatively sparse, and it would appear that in spite of a series of advantages, beyond sheer processing throughput, GPUs remain largely unexploited by ABM researchers. The juxtaposition of this latter dynamic and the high computational costs of large-scale ABMs leads to a dearth of such constructs. We conjecture that this is a constraint that must be ultimately removed, if ABMs are to deliver on their initial promise as a solution to problems that have proven intractable with traditional modelling tools. In this context, a valid observation is that neither of the aforementioned ABM specific frameworks is supportive of GPGPU acceleration at the time of this writing and, furthermore, the opportunities for even basic parallel acceleration are limited, in spite of attempts of retro-fitting it:

1. MASON (uses JAVA) can spawn multiple parallel schedulers that can execute concurrently, however it is most difficult to ensure race-free execution, thus leading do difficult to debug errors – no provisions for GPGPU or massive parallelism are present;
2. the Repast developers have opted for providing a separate version focused on high-performance parallel computing named Repast HPC (uses C++) – adoption rates have been low, because it is more difficult to use by non-expert programmers (most synchronization is directly under modeller control) – as was the case with MASON, GPGPU support is absent;
3. NetLogo (uses Scala and Java) is purely serial.

This article pursues two concurrent goals: investigating approaches to using GPGPU to accelerate ABM presented in the literature, and proposing an alternate, potentially superior approach to using GPUs for ABM in general, and consumer behaviour research in particular. We note that the latter is more demanding on the researchers, mandating an inter-disciplinary approach, but is likely to yield more comprehensive benefits in the long-run. Clearly, it would be folly to aim for an all encompassing analysis, as there is a breadth of applications of ABM in economics. As such, and as already hinted in the opening phrase, we focus on marketing specific models, primarily on those that study the dynamics of consumer behaviour.

We structure our work in the following manner:

- first, a non-exhaustive literature review is presented, aiming to analyse marketing specific ABM applications;
- to explore how to optimally exploit the parallel throughput offered by GPUs in the context of ABM based marketing research, we study the two main solutions demonstrated in the literature, which is to say either adhoc implementation of a model in one of the Application Programming Interfaces (API) currently available or using GPGPU accelerated ABM frameworks;
- we propose an alternative, more involved process in which the software implementation is developed by the researchers, but instead of directly using APIs and thus being forced into a low-level, non-portable paradigm, we focus on using high-level, hardware agnostic libraries; we argue that this latter approach is more supportive of novel research that cannot be easily cast within the confines of existing frameworks and, at the same time, tends to be conducted by non-expert programmers;
- in conclusion, we outline directions for future development and potentially optimal patterns for minimising friction induced by the uptake of ABM in marketing research in general and, more specifically, the use of GPGPU accelerated ABM.

It is our contention that this work, coupled with future developments, that will move from broad level overview into tangible implementations, will serve to "stoke" the fire under ABM based marketing research, thus making good on the initial promise of higher explanatory power made in the de facto manifest of ABM in social sciences due to Joshua Epstein and Rob Axtell[21].

2. Literature Review

It can be argued that Thomas Schelling's seminal work[22], which is generally accepted as the literary birth of ABM in social sciences, does include some marketing specific work in its 3rd chapter, wherein Akerlof's investigation into the market for “lemons”[23] is considered from a different, heterogeneous agent angle. Another important strand of research, that had notable impact on subsequent works, was that conducted into the allocative efficiency of markets when the traders are so called Zero-intelligence agents[24]. The latter serves as an early outline of the power of ABM, and in the sorts of insights that can be gained from conducting such analyses, as it shows that high market efficiency can be achieved even in the absence of elaborate behaviour such as active pursuit of optimality through inter-period optimisation. Further in the same vein, ABM's applicability to the study of complex phenomena such as the self-organization property of various markets has been researched as far back as the early 1990s[25].

Two data-points serve to illustrate the benefits of ABM in the analysis of consumer behaviour. Reference[26] studies the key phenomenon of “lock-in”, in which supremacy on a particular market is achieved by a single competitor, making it difficult for consumers to switch to another product. Contrary to Zero-intelligence experiments, agents are endowed with psychologically plausible decision processes, derived from behavioural theories such as those of social comparison and imitation. Such traits would be difficult to embed in a traditional mathematical model, although they do come at an increased computational cost, which might be one of the reasons for which a population of only 900 agents is used in the simulations.

Reference[27] is a relatively recent, in-depth into look at consumer behaviour through the ABM lens, from a different angle, that of the impact of quality variability on consumers' confidence, and, by extension, its potential for seriously damaging a market. It is also worth noting that the social network is taken into account (structure varies in accordance to multiple parametrizations spanning the two extremes of full connectivity and lack of connectivity), and identified as a primary determinant of market stability. The authors give substance to a frequently encountered, but seldom quantitatively supported claim in marketing science, that of the tree-like propagation of consumer experiences (e.g. a single unsatisfied customer generates 15 / 20 / 25 lost sales etc.). Thus, it is shown that aggregating group experience through social information leads to more balanced assessments and avoids the over-biasing due to singular bad experiences. This is yet again a sort of analysis that would be difficult to cast into a traditional modelling mould, as there is little room for such heterogeneity and granularity within the confines of an aggregate mathematical construct.

It is opportune to also point out that the Morgan Kaufmann's long-running and widely appreciated “Handbooks in Economics” series includes a volume entirely dedicated to ABM[28], which dedicates ample spaces to discussing topics most relevant to marketing such as the modelling of learning processes[29], endogenous interactions[30] or market design[31].

Table 1. *Some marketing research application areas suitable for ABM, reproduced after[10]*

Area	Advantage of ABM
Diffusion of information and innovations	Allows individual-level heterogeneity within both adoption decisions and social networks.
Retail location decisions.	Enables integration of individual-level behaviour patterns with geographic information systems.
Inter-firm relationships, strategy and competition.	Facilitates as many firms as necessary, each firm with firm-level characteristics, and the ability for firms to adapt their strategies over time
Marketing mix models	Allows the examination of individual-level behaviour patterns and reactions to the various elements of the marketing mix.
Retail and servicescape design.	Can be used to model individuals moving about and making decisions in a complex retail environment.

Possibly the most impressive use of ABM is embodied in EURACE[32], which is, to the authors' best knowledge, the largest-scale ABM constructed, and aims to represent in extensive detail the economy of the European Union. Three types of learning agents are included (households, firms and banks), interacting through five types of markets (consumption goods, investment goods, labour, credit and financial assets). Agent counts are orders of magnitude higher than what is commonly present in the literature (e.g. the upper bound on household count is 107). EURACE is intended to be holistic, and to study economic interactions as they occur in the context of an over-arching, inclusive system. This clearly presents novel opportunities for marketing researchers, who could leverage findings from multiple threads of investigation and organically integrate them in the same analytical tool. A limitation of EURACE was that in its fully fledged form it relied on parallel acceleration through large-scale computing clusters (local or distributed), which limits its usability to researchers that can access such constructs. In theory, GPGPU acceleration of ABM would lead to such large-scale models becoming feasible on local machines equipped with the right (comparatively inexpensive) hardware, thus making explorations in this directions even more opportune if not outright mandatory. Given spatial constraints, we must limit the more verbose analysis, however we will point the interested reader towards a recent, comprehensive cross-section of the topic, due to Rand and Rust[10]. We reproduce below, in Table 1, the abbreviated synopsis of marketing research areas suitable for ABM:

3. ABM and GPGPU

An in-depth discussion of programming or, more specifically, GPGPU, is beyond the scope of this work. We direct the interested reader to Chapter 4 from[16] for an in extenso treatise of the hardware architecture and to[33],[34] for programming paradigms and directions for the software side. In a nutshell, GPUs are many-core machines (e.g. up to 32 cores on a modern high-end processor[35]), which use relatively wide (i.e. 32 or 64 elements in the two dominant GPU architectures) Single Instruction Multiple Data (SIMD[36]) execution on vector units attached to each core to achieve high computational density, and implicitly throughput. High-end GPUs are also endowed with high-speed memory attached to wide buses, thus offering an order of magnitude higher memory bandwidth than most CPUs. Finally, in spite of the very impressive theoretical performance (e.g. 4 single-precision TeraFLOPs), GPUs are available as consumer devices, and thus both widely available and competitively priced. This serves to explain the explosive interest in levering them as general-purpose programmable devices, capable of accelerating a wide array of scientific applications.

Table 2. *ABMs using GPGPU[10]*

Reference	ABM implemented	API / Framework
[40]	SugarScape (simplified), StupidModel.	OpenGL.
[41]	Mood Diffusion, Conway's Game of Life.	OpenGL.
[42]	Crowd simulation.	OpenGL.
[43]	Crowd simulation.	Direct3D.
[44]	Crowd simulation.	CUDA.
[45]	Crowd simulation.	CUDA.
[46]	Crowd simulation.	CUDA.
[47]	Crowd simulation.	OpenGL, CUDA.
[48]	Conway's Game of Life, Leadership Model (LDR).	CUDA.
[49]	Spatial opinion diffusion.	CUDA.

Parallel programming is generally accepted as a challenging task, with an extensive span of specific challenges that are not apparent when writing serial code. In the case of GPUs, this is compounded with the heterogeneity of the system, as the GPU is a separate processing platform, an accelerator, that has its own discrete memory and executes asynchronously. The fact that GPUs employ wide-SIMDs serves to further raise the challenge, as the extraction of optimal performance implies consideration for control flow coherence or independent execution across multiple elements. In order to tackle this difficulty, most if not all GPU compute APIs, such as CUDA, OpenCL or C++ AMP, follow a Single Instruction Multiple Thread (SIMT) or Single Program Multiple Data (SPMD) paradigm, abstracting away much of the complexity[37],[38],[39]. We will not delve into the differences between the above listed 3-tuple, beyond noting that on a basic level all can be used with success and with varying degrees of difficulty, as is evidenced by the ever-growing literature on GPGPU. We note, however, that outside of the scope of numerical work such as linear algebra or image processing, success in using GPGPU has been limited. By extension, this also applies to ABM. In Table 2, we list the main cases of GPGPU use in ABM literature:

It is easy to observe that the scope of the implemented models is rather narrow, and that there is no exploratory modelling work. Whilst this observation is not intended to diminish the merits of the research into optimal parallel implementations of existing models, it is clear that there is a rather noticeable gap by comparison with the complexity of the models we listed in section 2. Indeed, in certain cases the number of agents is up to three orders of magnitude higher in the GPGPU accelerated case, but these are too simple agents engaging in too simple interactions. Excluding the case of the zero-intelligence experiments, it appears that the needs of marketing researchers are notably more complex in what regards agent specification. We outline what we consider to be the two causes for this discrepancy:

1. Programming GPUs is challenging: it is unlikely that non-specialists, researchers without a specific background in Computer Science, are going to invest the time into acquiring a skill that is not certain to augment their chances for publications and directly support their core work; in this context, the use of dedicated frameworks is more appealing, as it allows the (economics, marketing) specialist to focus on areas where his expertise lies, without having to fight what seems like an uphill programming battle;
2. Many data-structures (e.g. linked lists) and approaches commonly used by serial ABMs (e.g. centralized pseudo-random number generation) do not map well, if at all, to the GPU; moreover, at this time, GPUs support only a restricted subset of the full set of features available on CPUs (e.g. no dynamic allocation / deallocation of memory, no runtime polymorphism) it is therefore impossible to merely “pilfer” existing code, and new code has to be written, however the first constraint makes that a seldom feasible solution.

Clearly, the goal should be a feasible combination of the usability of existing and widely used ABM frameworks, and the throughput made possible by levering GPUs. Otherwise, and somewhat more prosaically stated, we aim to have (at least some of) the new “free lunch” afforded by the sustained growth in parallel throughput, whilst abstracting the latter's complexity to the greatest extent possible. We envision two approaches capable of reaching this goal:

1. New frameworks are built by 3rd parties that have the programming knowledge required;
2. Higher-level libraries are used by the researcher(s) in order to implement a particular model.

Out of the two approaches, we expect the first to serve as the main vehicle for expanding the uptake of GPGPU accelerated ABM by marketing researchers who are not keen on programming (historically, this has been the case for serial ABMs too). The second approach, whilst decidedly more involved, is also the one that yields maximal flexibility and thus is the most supportive of highly exploratory work – models that do not fit well within what is expressible with a general-purpose framework can only be analysed by manually implementing them. In what follows, we will detail both cases, GPGPU accelerated ABM frameworks as represented by FLAMEGPU[50], as well as autonomous development and implementation, based on our experiences with the latter.

4. The FLAMEGPU Framework

FLAMEGPU is the moniker under which GPU acceleration is being added to the FLAME framework. The initial goal for FLAME was to support molecular / cellular level modelling as required by, for example, computational biologists. It has grown into a generic ABM framework and has been used with some success outside of its initial scope, being, quite notably as the baseline for the EURACE implementation.

At the time of this writing, FLAME uses NVIDIA's proprietary CUDA API to access the GPU, thus binding it to the company's hardware. Work on an OpenCL implementation is on-going, so it can be theorised that at some point in the future the vendor lock-in will be removed. Whilst there are valid reasons for using CUDA (e.g. optimal exposure of the hardware, mature tool-chain), we consider the requirement for a particular brand of GPU an unfortunate constraint that limits the usability of FLAMEGPU to some extent.

Researchers are not exposed to these low-level details (unless they choose to): the simulation code is generated based on specification written by the modeller in XML with extendible schemas. Agents are formally specified as state machines with internal memory (dubbed X-Machines). Message passing and behaviour are defined through functions that rely on state. A set of XSLT code templates are used, in conjunction with the model specified in XML, during the parse phase in order to generate compilable simulation code that is linkable against C based agent function scripts. From the perspective of the researcher, the latter steps tend to be opaque, therefore insulating him from programming details he may have no interest in. An exhaustive analysis of FLAME's inner workings cannot be fit into the space allocated for this article, but fortunately this has already been presented in the literature[51]. It is important to note that the performance increases yielded by the use of GPUs are very high[50],[51], although we must remain wary of potential sub-optimality in the CPU baseline, which is rather frequently encountered in such analyses[52].

To the best of the authors' knowledge, no marketing focused ABMs have been implemented using FLAME in general of FLAMEGPU in particular, therefore it is difficult to forecast how successful such an endeavour would be. We note that specifying a model in FLAME is still somewhat more involved than in, for example, NetLogo, which might serve as a mild deterrent. Moreover, knowledge of its existence and features is still scarce, if at all existent, within the marketing community. Extensive experimentation will be needed before the match between the needs of the marketing researcher and the facilities offered by FLAME can be gauged.

5. Developing GPU Accelerated ABMs

As evidenced in Table 2, on-the-spot implementation of ABMs in a GPU API has constituted the main conduit for exploiting this new vein of computational performance. However, all the enumerated works (excluding the ones that use FLAME, which was discussed in section 4) opt for a rather low level of abstraction, remaining too close to the hardware (e.g. use of primitive containers such as C arrays, manual implementations of primitives such as parallel prefix scan / sum). In our work we have found this to yield an increase in complexity and a decrease of generic power, without major benefits in what regards performance (the main claimed benefit of avoiding abstraction). In effect, we conjecture that the optimal approach to the manual implementation of ABMs must rely on higher level abstractions, such as the containers and algorithms included in the C++ standard library[53]. When new Abstract Data Types (ADTs) or algorithms need to be put together, we align ourselves with the principles expressed in[54], and seek to draw out the most generic characteristics, as opposed to tailoring solutions to a single application / problem. For this we have found the C++ programming language, and especially its template mechanism[55] to be a particularly good fit, and have used it extensively.

Undoubtedly, at this point it becomes important to point out that a GPU accelerated equivalent of the C++ standard library, or, more specifically, its full complement of containers and algorithms, is not available. The closest approximation is probably the Thrust library[56], which is NVIDIA specific. A second, notably less developed, option is represented by the Bolt library[57], which has the advantage of vendor agnosticism. We note that even the latter, less fleshed out library, offers all the “ingredients” required by the GPGPU accelerated ABMs from Table 2, and it is straightforward to port them. We will not detail how such a port could be carried out, as we deem such an exercise merely interesting practice, but rather explore how one manually implement an ABM for studying consumer behaviour using C++ AMP. The latter represents a library and minimal language extension of the C++ language that makes GPUs accessible with minimal friction. We use C++ AMP as it supports all the C++ language constructs and idioms we require, and is aligned with the latest C++11 standard.

Let us start from the skeleton of the ABM presented in[58], which we modify to consider the propagation of product perception and, implicitly, purchase behaviour. In good ABM tradition, our construction commences from the bottom up, with the agents, in our case consumers: We represent them through an ADT, in effect a class template:

```
template<typename Decision_rule, Neighbourhood N>class Consumer { // Member functions and data. };        (1)
```

Through the Decision_rule type parameter we parametrize the agent's behaviour, going from simple, zero-intelligence decisions to elaborate ones based on rigorous models such as the Bush-Mosteller[59]. The Neighbourhood ADT defines the extent of the agent's interaction area and can expose it to queries through public member functions. Obviously, this layout also supports heterogeneity amongst agents, as nothing prevents a mix of agents from co-existing, with differing decision rules and neighbourhoods (e.g. leaders / trend setters might have a wider neighbourhood and more elaborate decision making). Of course, since we use compile-time polymorphism through the use of templates, which leads to different parametrizations being, in effect, different types as far as C++ is concerned, a heterogeneous population would require the use of multiple containers, but that hardly poses a major problem. All that we require of our Consumer ADT is that it exposes a public update() member function, and a set of public member functions that provide external, non-mutable visibility of agent's state. We use the former to update the state of the simulation within each iteration, and the latter to compute various statistic quantities necessary for deriving insight from the simulation. The agent populations are packaged in std::vector<Consumer<DR, N>> containers[53], where DR stands for class matching the Decision_rule constraints, and N plays the similar role for a class matching the Neighbourhood constraints. To pass them to the GPU for processing, we wrap the containers in instantiations of the array_view class template[39], and then schedule parallel execution for the entire set of agents.

For brevity, we defer going into in-depth programming details for an upcoming, yet unpublished work. However, it is difficult to regard the traversal of the above listed steps as overly challenging. Still, we have found that it is optimal to decouple model conception with its effective implementation. The former phase is better entrusted to dedicated marketing researchers, with limited input from computational economists to prevent the elaboration of models that are outright infeasible. On the contrary, the implementation phase should be undertaken by computational economists. Correctness checking and the actual experiments are, as is easy to conclude, a mixed effort. In closing, we note that if the abstraction level is kept high, the resulting ADTs and algorithms can be reused across a wide span of models – our Consumer class template can also be used to instantiate agents in Sugarscape, or other ABMs.

6. Conclusions

In this paper, we have conducted a broad-level analysis of the topic of ABM uses in marketing research and the emerging field of GPGPU acceleration of such models. Whilst there are a number of applications of ABM in marketing presented in the literature, we could find no works that considered the problem of performance improvement through use of parallel programming in general or, GPU programming in particular. We identified this as a weakness, as the expectation of sustained single-processor performance progression is no longer aligned with current realities. In this context, we have identified two paths for leveraging the parallel throughput of modern GPUs to accelerate ABM: using a framework, in this case FLAMEGPU, or manually implementing the model by using higher level abstractions and libraries. For the latter case we provide an overview of how the process could be carried out by inter-disciplinary teams that include both marketing researchers and computational economists.

Considering the growing number of opinions that position ABM as the next big step in economics modelling, as well as the wide scope of applications that have already been published in the field of marketing, it would be most unfortunate if the potential for development is hampered by lack of computational support for larger-scale complex ABMs. As such, we hope that our high-level work will spawn interest within the marketing community for GPGPU usage to speed up ABM. These endeavours should be carried out in an interdisciplinary fashion. In our future works, we will detail the procedural aspects, as well as demonstrate a series of implementations, which we expect to provide consistent proof of ABM's merits in the context of marketing research, as well as supporting evidence for the approaches we studied in this paper.

Acknowledgements

This work was cofinaced from the European Social Fund through Sectoral Operational Programme Human Resources Development 2007-2013, project number POSDRU/107/1.5/S/77213 „Ph. D. for a career in interdisciplinary economic research at the European standards”.

References

[1] J. H. Holland, “Complex adaptive systems,” *Daedalus*, vol. 121, no. 1, pp. 17–30, 1992.

[2] K. J. Arrow and G. Debreu, “Existence of an equilibrium for a competitive economy,” *Econ. J. Econ. Soc.*, pp. 265–290, 1954.

[3] W. T. Bielby and R. M. Hauser, “Structural equation models,” *Annu. Rev. Sociol.*, vol. 3, pp. 137–161, 1977.

[4] J. W. Forrester, *Industrial dynamics*, vol. 2. MIT press Cambridge, MA, 1961.

[5] M. Buchanan, “Economics: meltdown modelling,” *Nature*,

vol. 460, no. 7256, pp. 680–682, 2009.

[6] J. D. Farmer and D. Foley, "The economy needs agent-based modelling," *Nature*, vol. 460, no. 7256, pp. 685–686, 2009.

[7] D. Colander, M. Goldberg, A. Haas, K. Juselius, A. Kirman, T. Lux, and B. Sloth, "The Financial Crisis and the Systemic Failure of the Economics Profession," *Crit. Rev.*, vol. 21, no. 2–3, pp. 249–267, 2009.

[8] D. Colander, "The Failure of Economists to Account for Complexity," *Causes of the Crisis*, 12-Sep-2009. .

[9] G. E. Moore, *Cramming more components onto integrated circuits*. McGraw-Hill, 1965.

[10] W. Rand and R. T. Rust, "Agent-based modeling in marketing: Guidelines for rigor," *Int. J. Res. Mark.*, vol. 28, no. 3, pp. 181–193, 2011.

[11] R. Axtell, "Why agents? On the varied motivations for agent computing in the social sciences," 2000.

[12] K. L. Judd, "Chapter 17 Computationally Intensive Analyses in Economics," in in *Handbook of Computational Economics*, vol. Volume 2, Elsevier, 2006, pp. 881–893.

[13] "NetLogo User Manual (version 5.0.1)."[Online]. Available: http://ccl.northwestern.edu/netlogo/faq.html.[Accessed: 25-Jun-2012].

[14] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, and G. Balan, "MASON: A Multiagent Simulation Environment," *SIMULATION*, vol. 81, no. 7, pp. 517–527, Jul. 2005.

[15] M. J. North, T. R. Howe, N. T. Collier, and J. R. Vos, "The repast simphony runtime system," in *Proceedings of the Agent 2005 conference on generative social processes models and mechanisms*, 2005, pp. 151–158.

[16] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*. Elsevier, 2011.

[17] H. Sutter, "The free lunch is over: A fundamental turn toward concurrency in software," *Dr Dobb's J.*, vol. 30, no. 3, pp. 202–210, 2005.

[18] "Welcome to the Jungle," *Sutter's Mill*.[Online]. Available: http://herbsutter.com/welcome-to-the-jungle/.[Accessed: 09-May-2013].

[19] A. R. Brodtkorb, C. Dyken, T. R. Hagen, J. M. Hjelmervik, and O. O. Storaasli, "State-of-the-art in heterogeneous computing," *Sci. Program.*, vol. 18, no. 1, pp. 1–33, Mar. 2010.

[20] J. J. Dongarra and A. J. van der Steen, "High-Performance Computing Systems: Status and Outlook," *Acta Numer.*, vol. 21, pp. 379–474, 2012.

[21] J. M. Epstein and R. L. Axtell, *Growing Artificial Societies: Social Science from the Bottom Up*, First Edition. A Bradford Book, 1996.

[22] T. C. Schelling, *Micromotives and macrobehavior*. WW Norton & Company, 2006.

[23] G. A. Akerlof, "The market for' lemons': Quality uncertainty and the market mechanism," *Q. J. Econ.*, pp. 488–500, 1970.

[24] D. K. Gode and S. Sunder, "Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality," *J. Polit. Econ.*, pp. 119–137, 1993.

[25] N. J. Vriend, "Self-organization of markets: An example of a computational approach," *Comput. Econ.*, vol. 8, no. 3, pp. 205–231, 1995.

[26] M. Janssen and W. Jager, "An integrated approach to simulating behavioural processes: A case study of the lock-in of consumption patterns," *J. Artif. Soc. Soc. Simul.*, vol. 2, no. 2, pp. 1–29, 1999.

[27] S. S. Izquierdo and L. R. Izquierdo, "The impact of quality uncertainty without asymmetric information on market efficiency," *J. Bus. Res.*, vol. 60, no. 8, pp. 858–867, 2007.

[28] L. Tesfatsion and K. L. Judd, *Handbook of Computational Economics: Agent-Based Computational Economics*. Elsevier, 2006.

[29] T. Brenner, "Agent learning representation: Advice on modelling economic learning," *Handb. Comput. Econ.*, vol. 2, pp. 895–947, 2006.

[30] N. J. Vriend, "ACE models of endogenous interactions," *Handb. Comput. Econ.*, vol. 2, pp. 1047–1079, 2006.

[31] R. Marks, "Market design using agent-based models," *Handb. Comput. Econ.*, vol. 2, pp. 1339–1380, 2006.

[32] C. Deissenberg, S. van der Hoog, and H. Dawid, "EURACE: A massively parallel agent-based model of the European economy," *Appl. Math. Comput.*, vol. 204, no. 2, pp. 541–552, Oct. 2008.

[33] V. Pallipuram, M. Bhuiyan, and M. Smith, "A comparative study of GPU programming models and architectures using neural networks," *J. Supercomput.*, pp. 1–46.

[34] B. R. Gaster and L. Howes, "FUNDAMENTAL PROBLEMS," 2012.

[35] AMD, "AMD Graphics Core Next (GCN) Architecture." AMD, 28-Jun-2012.

[36] M. J. Flynn, "Some Computer Organizations and Their Effectiveness," *Ieee Trans. Comput.*, vol. C–21, no. 9, pp. 948–960, 1972.

[37] A. Habermaier and A. Knapp, "On the correctness of the SIMT execution model of GPUs," *Program. Lang. Syst.*, pp. 316–335, 2012.

[38] I. Buck, "GPU computing with NVIDIA CUDA," in *ACM SIGGRAPH 2007 courses*, New York, NY, USA, 2007.

[39] B. Gaster, D. R. Kaeli, L. Howes, P. Mistry, and D. Schaa, *Heterogeneous Computing with OpenCL*. Morgan Kaufmann, 2011.

[40] K. Gregory and A. Miller, *C++ AMP: Accelerated Massive Parallelism with Microsoft Visual C++*. Microsoft Press, 2012.

[41] M. Lysenko and R. M. D'Souza, "A framework for megascale agent based model simulations on graphics processing units," *J. Artif. Soc. Soc. Simul.*, vol. 11, no. 4, p. 10, 2008.

[42] K. S. Perumalla and B. G. Aaby, "Data parallel execution challenges and runtime performance of agent simulations on GPUs," in *Proceedings of the 2008 Spring simulation multiconference*, San Diego, CA, USA, 2008, pp. 116–123.

[43] P. Richmond and D. M. Romano, “A high performance framework for agent based pedestrian dynamics on gpu hardware,” *Proc. Eurosis Esm*, 2008.

[44] J. Shopf, J. Barczak, C. Oat, and N. Tatarchuk, “March of the Froblins: simulation and rendering massive crowds of intelligent and detailed creatures on GPU,” in *ACM SIGGRAPH 2008 classes*, New York, NY, USA, 2008, pp. 52–101.

[45] H. Li, A. Kolpas, L. Petzold, and J. Moehlis, “Parallel simulation for a fish schooling model on a general-purpose graphics processing unit,” *Concurr. Comput. Pr. Exp.*, vol. 21, no. 6, pp. 725–737, 2009.

[46] U. Erra, B. Frola, V. Scarano, and I. Couzin, “An Efficient GPU Implementation for Large Scale Individual-Based Simulation of Collective Behavior,” in *International Workshop on High Performance Computational Systems Biology, 2009. HIBI ’09*, 2009, pp. 51–58.

[47] E. B. Passos, M. Joselli, M. Zamith, E. W. G. Clua, A. Montenegro, A. Conci, and B. Feijo, “A bidimensional data structure and spatial optimization for supermassive crowd simulation on GPU,” *Comput Entertain*, vol. 7, no. 4, pp. 60:1–60:15, Jan. 2010.

[48] A. R. D. Silva, W. S. Lages, and L. Chaimowicz, “Boids that see: Using self-occlusion for simulating large groups on GPUs,” *Comput Entertain*, vol. 7, no. 4, pp. 51:1–51:20, Jan. 2010.

[49] B. G. Aaby, K. S. Perumalla, and S. K. Seal, “Efficient simulation of agent-based models on multi-GPU and multi-core clusters,” in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, ICST, Brussels, Belgium, Belgium, 2010, pp. 29:1–29:10.

[50] W. Tang and D. A. Bennett, “Parallel agent-based modeling of spatial opinion diffusion accelerated using graphics processing units,” *Ecol. Model.*, vol. 222, no. 19, pp. 3605–3615, Oct. 2011.