Entwicklungsumgebungen für Multi-Agenten-Systeme zur Simulation von digitalen Marktplätzen

MICHAEL CLASEN, KIEL

Abstract

The latest developments in the field of electronic markets have shown, that knowledge about success factors is limited, making further research necessary. The small number of existing market places does not allow empiric research with significant results and experiments in social sciences are hardly realizable. Therefore analysis with multi-agent-simulation (MAS) could be a usable method. In this article two MAS-Workbenches StarLogo and Ascape are compared to constructing the simulation with the computer language C.

1. Einführung

Seit einiger Zeit werden Multi-Agenten-Systeme (MAS) mit wachsendem Erfolg eingesetzt, um Problemstellungen der unterschiedlichsten Forschungsbereiche zu simulieren (Rauch 2002). Zu den bekanntesten Arbeiten zählen sicherlich die Sugarscape-Modelle von Epstein und Axtell (1996), mit denen eine Vielzahl sozialer Phänomene nachgebildet und z.T. erklärt werden konnten. Aber auch im Bereich der Agrarwissenschaften finden sich vermehrt Arbeiten, in denen Multi-Agenten-Simulationen mit Erfolg eingesetzt worden sind (z.B. Berger 2000; Balmann 1995).

Für eine stärkere Verbreitung von MAS in der wissenschaftliche Praxis ist es notwendig, dem Forscher leistungsfähige und komfortabel zu bedienende Werkzeuge zur Erstellung und Ausführung der Simulationen zur Verfügung zu stellen. StarLogo und Ascape stellen solche Entwicklungsumgebungen für MAS dar, und werden in diesem Beitrag bzgl. Nutzen und Einarbeitungsaufwand mit einer vollständig proprietären Programmierung in C verglichen.

Um möglichst praxisnahe Aussagen machen zu können, wurden beide Entwicklungsumgebungen anhand einer konkreten Simulationsaufgabe getestet. Die Aufgabe bestand darin, eine Multi-Agenten-Simulation zu erstellen, mit welcher Einsichten über das Entstehen und Verschwinden der neuen digitalen Marktplätze in der Agrar- und Ernährungswirtschaft gewonnen werden können.

2. Modellierungsaufgabe: Simulation von digitalen Marktplätzen

Für den hier durchgeführten Vergleich von Simulationswerkzeugen wird folgendes sehr einfaches Modell angenommen. Auf einer Simulationsfläche werden n Landwirte und m Marktplätze zufällig verteilt. Landwirte ernten in jeder Periode t eine zufällig variierende Menge an Getreide und verkaufen sie auf dem Marktplatz, bei dem die geringsten Gesamtkosten entstehen. Die Gesamtkosten einer Handelstransaktion setzen sich aus den Transportkosten zwischen Landwirt und Marktplatz, Nichtvertrauenskosten und Liquiditätskosten zusammen. Die Transportkosten sind während der Simulationsdauer konstant, während sich Nichtvertrauenskosten und Liquiditätskosten nach festgelegten Regeln verändern, was zu Pfadabhängigkeiten führen kann.

Anhand dieses einfachen Modells, kann nun versucht werden zu simulieren, wie viele und welche der digitalen oder konventionellen Marktplätze im Zeitablauf bei bestimmten Ausgangsparametern existieren können.

Im folgenden soll diskutiert werden, ob sich StarLogo oder Ascape zur Simulation dieser Art von Modellen eignet oder, ob eine proprietäre Programmierung in einer Hochsprache vorzuziehen ist.

3. Simulationsumgebungen

Die Vorteile von Simulationsumgebungen gegenüber konventionellen Computersprachen liegen hauptsächlich in der leichteren Erlern- und Anwendbarkeit, sowie der graphischen Ausgabe der Ergebnisdaten. Diesen Vorteilen stehen häufig eingeschränkte Möglichkeiten und ein zum Teil enormer Performance-Verlust gegenüber.
Ob eine Simulationsumgebung leicht zu erlernen ist, hängt zum einen von der Vorbildung des Anwenders und zum anderen vom Vorhandensein von Handbüchern, Tutorien, Beispielen, Newsgroups, Diskussionsforen und dem Quellcode der Simulationsumgebung ab. Bei der Anwendbarkeit kommt es darauf an, wie schnell ein Modell in ein Programm umgesetzt werden kann. Da die Programmierung von Simulationsmodellen häufig exploratorisch ist, ist eine schnelle Änderbarkeit des Codings und ein einfaches Debugging besonders wichtig (Gilbert und Troitzsch 1999, S. 20). Die große Fülle an generierten Daten erfordert es zudem, das Datenmaterial möglichst noch während der Laufzeit graphisch aufzubereiten, um die Veränderungen schon während der Simulation beobachten zu können.
Inwieweit diese prinzipiellen Vorteile von Simulationsumgebungen bei StarLogo und Ascape umgesetzt worden sind, und auf was der Anwender verzichten muss, wird im folgenden näher betrachtet.

3.1 StarLogo

StarLogo (http://www.media.mit.edu/StarLogo/) basiert auf der schon in den 60er Jahren entwickelten Programmiersprache Logo, die bewusst einfach gestaltet wurde, um auch Schüler an die Programmierung von Computern heranzuführen. Auffälligstes Element von Logo ist die sogenannte Turtle-Graphik, bei der der Programmierer eine Schildkröte mit einfachen Befehlen auf dem Bildschirm bewegt und dadurch Graphiken erzeugen kann.
Für die Simulation von Multi-Agenten-Systemen verwendet StarLogo nicht mehr nur eine Schildkröte, sondern beliebig viele, die weiterhin mit einfachen Befehlen dirigiert werden können.
Wer ein wenig Programmiererfahrung mitbringt, kommt mit Hilfe der guten und umfangreichen Handbücher und der Vielzahl an guten, leicht nachzuvollziehenden Beispielen schnell mit StarLogo zurecht und kann schon nach wenigen Stunden eigene Simulationen erstellen. Für kniffligere Probleme steht zudem ein gut besuchtes Informationsforum zur Verfügung, in dem u.a. auch die Entwickler von StarLogo innerhalb weniger Tage jede Frage beantworten.
StarLogo besteht aus vier Fenstern, von denen eines der Programmierung, eines der Steuerung und zwei der Ausgabe der Daten dienen. Eigene Programme können mit sehr wenigen Zeilen Code erstellt werden, da eine Vielzahl zum Teil sehr spezieller Befehle zur Verfügung stehen. Aufgrund dieser einfachen Handhabung wird auch StarLogo wie sein Vorgänger Logo in Schulen zur Visualisierung von dezentralen Systemen eingesetzt.
Die Ausgabe der Ergebnisse erfolgt permanent während der Simulation über das Hauptfenster, über einen Graphen, in dem beliebige Zustände der Turtles als Kurve aufgezeichnet werden und über ein Textfenster, in dem Daten als ASCII-Zeichen ausgegeben werden können.
Da StarLogo in Java programmiert ist, ist es auf nahezu allen Plattformen lauffähig; der Nachteil liegt jedoch in einer schlechten Performance.
StarLogo eignet sich hervorragend um Agenten zu beobachten, die sich im Raum unter bestimmten Restriktionen bewegen bzw. verteilen. An seine Grenzen stößt StarLogo, wenn unterschiedliche Agenten in komplexer Weise miteinander in Kontakt treten und miteinander

kommunizieren müssen. Da z.B. keine Arrays unterstützt werden, ist das Arbeiten mit größeren Datenmengen nur sehr eingeschränkt möglich.
Die neueste Weiterentwicklung von StarLogo heißt NetLogo und ist im Release 1.0 seit April 2002 unter folgender Adresse zu beziehen: http://ccl.northwestern.edu/netlogo/. Neben einigen erweiterten Programmiermöglichkeiten liegt der Hauptunterschied zu StarLogo in der Möglichkeit, die Simulationen als Java-Applet in einem Browser ablaufen zu lassen. Hierdurch ist es möglich, verteilte Simulationen z.B. über das Internet ablaufen zu lassen, in dem jeder Teilnehmer einen oder mehrere Agenten steuert. In diesem Falle wird die künstliche Intelligenz herkömmlicher Agenten durch die natürliche Intelligenz der Mitspieler ersetzt.

3.1 Ascape

Ascape (http://www.brook.edu/dybdocroot/es/dynamics/models/ascape/) erhebt den Anspruch, ähnlich komfortabel wie StarLogo, jedoch deutlich mächtiger zu sein, so dass auch komplexe Modelle dargestellt werden können. Die graphische Darstellung der Daten ist übersichtlich und bietet einige zusätzliche Möglichkeiten, wie die Darstellung als Histogramm oder Tortendiagramm. Auch die Bedienung des Modells erfolgt intuitiv über eine Kontrollleiste.
Die Erstellung von neuen Modellen erweist sich allerdings als deutlich schwieriger als unter StarLogo, da neue Simulationen „hart“ in Java codiert werden müssen. Ascape, welches ebenfalls in Java geschrieben worden ist, stellt hierzu „lediglich“ allgemeine Klassen und Methoden zur Darstellung, Bewegung und Steuerung von Agenten in einem zellulären Automaten und zur Ausgabe der Daten bereit. Um neue Ascape-Simulationen erstellen zu können, sind also fundierte Java-Kenntnisse notwendig. Erschwert wird das Vorhaben durch eine unzureichende Dokumentation. Die Diskussionsforen sind ähnlich gut besucht und hilfreich wie bei StarLogo.

4. Eignung für die Simulation von digitalen Marktplätzen

Beim Versuch, das oben beschriebene Marktplatzmodell mit StarLogo zu erstellen, wurden schnell die Grenzen der Möglichkeiten erreicht. Speziell der Bedarf, eine Vielzahl an Zuständen der Agenten (Landwirte und Marktplätze) in geeigneten Datenstrukturen abzuspeichern, gelang mit StarLogo nicht oder nur über unzufriedenstellende Kunstgriffe. Des weiteren musste festgestellt werden, dass die besonderen Fähigkeiten zellulärer Automaten (Mueller 1995, S. 2), Bewegungen der Agenten im Raum in Echtzeit darzustellen, nicht genutzt werden konnten, da in dem oben vorgestellten Modell die Agenten „Marktplatz“ und „Landwirt“ im Raum statisch platziert wurden.
Vor diesem Hintergrund, die meisten Features der Simulationsumgebungen bei der Modellierung von Marktplätzen nicht nutzen zu können, wurde von einer Realisierung mit Ascape, obwohl sicherlich möglich, bisher abgesehen. Stattdessen wurde das Modell zunächst in C implementiert. Die Vorteile der Realisierung mit C liegen, neben der größeren Vertrautheit des Autors mit C, in einer deutlich besseren Performance, der größeren Übersichtlichkeit des Codings und der größeren Nähe zu den Daten. In einem vollständig selbst geschriebenen Programm gibt es keine „black-boxes“, die dem Wissenschaftler verborgen bleiben.
Der Nachteil einer komplett selbst geschriebenen Lösung liegt in der nicht vorhandenen Aufbereitung der Ergebnisdaten. Gelöst wurde dieses Problem, indem während der Laufzeit die relevanten Daten in ein ASCII-File geschrieben wurden und anschließend (zum Teil per Excel-Makro) in Excel ausgewertet und graphisch dargestellt wurden. Bis auf die Darstellung der Daten während der Laufzeit ist auf diese Weise jede erdenkliche Auswertung möglich.

5. Zusammenfassung

Die hier dargestellten Entwicklungsumgebungen StarLogo und Ascape dienen der Programmierung zellulärer Automaten. Ihre Vorteile können die Werkzeuge daher nur dann voll ausspielen, wenn sich Agenten während der Simulation im Raum bewegen und diese Bewegungen Untersuchungsgegenstand sind.
StarLogo bietet einen einfachen und schnellen Einstieg in die Welt der Multi-Agenten-Simulation. Leichte bis mittelschwere Probleme sind nach kurzer Einarbeitungsphase schnell zu verwirklichen. Für komplexere Probleme, an denen StarLogo scheitert, bietet sich Ascape an. Durch die Verwendung reinen Java-Codings zur Beschreibung der Agenten und ihrer Verhaltensregeln bietet Ascape alle Möglichkeiten, die auch Java bietet.
Für Wissenschaftler, die Java nicht beherrschen, stellt dies jedoch eine hohe Einstiegsbarriere dar. Somit stellt die Forderung nach hoher Mächtigkeit und einfacher Erlernbarkeit einer Entwicklungsumgebung für Multi-Agenten-Systeme weiterhin einen trade-off dar.
Das Erlernen einer anspruchvollen Umgebung wie Ascape lohnt sich daher wohl nur, wenn häufig Simulationen durchgeführt werden müssen.

6. Literatur

Balmann, A. (1995): Pfadabhängigkeiten in Agrarstrukturentwicklungen – Begriff, Ursachen und Konsequenzen, Berlin.

Berger, T. (2000): Agentenbasierte räumliche Simulationsmodelle in der Landwirtschaft – Anwendungsmöglichkeiten zur Bewertung von Diffusionsprozessen, Ressourcennutzung und Politikoptionen, Agrarwirtschaft, Sonderheft 168, Bergen/Dumme

Epstein, J. M., Axtell, R. (1996): Growing Artificial Societies. Social Science from the Bottom up, Washington.

Gilbert, N.; Troitzsch, K. G. (1999): Simulation for the Social Scientist, Buckingham, Open University Press

Mueller, R. A. E. (1995): Artificial Life: Frequently asked Questions (FAQS) and partially baked Ideas (PBIS) about its Uses in Agricultural Research, 2nd IFAC/IFIP/EurAgEng Workshop on Artificial Intelligence in Agricultural in Wageningen, 29-31 May 1995

Rauch, J. (2002): Seeing Around Corners, The Atlantic Online, http://www.theatlantic.com/issues/2002/04/rauch.htm, Stand: 13.06.02