人工知能を使った先物市場実験

中央大学商学部金融学科　小林　重人

はじめに

私が文系学生であり、そしてプログラミング経験が皆無であったにもかかわらず、コンピュータを使用した市場実験を行うまでに至ったのは商学の枠を大きく超えて、多くの人たちと出会ったからに他ならない。それまで私は研究というものが確固たる理論の上に議論が重なっていくことだけで生まれていくものであると考えていた。しかし、ある理論を実践するため、または新たな発見を探求するために自ら環境を作り出してしまった工学系の人々に出会ったとき、私のその考えはかなり揺らいだ。もはや椅子に座って机に向かっているだけでは私の横を光速の勢いで通り過ぎていく魅力ある新しいものを取り逃がしてしまうと。ましてや工学系の人たちが先物市場を設計し、そして完成させたという事実も私を驚愕させた。このままでは文系が主流であるとされている分野が工学系に席捲されてしまう。まさか、そのようなことが起こりうるはずがない。自信を持って言える状況であるとはとても思えなかった。そのとき感じたのは、これからの時代に文系、理系にこだわっていたのでは危ういのではないかということである。多くの工学系の人たちだけでこのようなものが作れるのであれば、領域が違うとされる分野の人々と合わさって研究を行ったらどうなるのであろうか。そこに私は何か閾値を越えたときに起こる爆発的なエネルギーのようなものがあるのではないかと大きく感じた。よいものがあれば既存の住み分けでは自分にはあてはまらないというものでも吸収していく。そしてどんどん学び、取りいれていくことが新時代に必要な研究スタイルだと最近考えている。

これは Richard Burton（Duke University's Fuqua School of Business）の言葉であるが、これからは文系と理系の垣根を超えた**ドッキング・ラボラトリー（Docking Laboratory）**が重要である。つまり、あるひとつのカテゴリーにとらわれていたのでは、これからの研究は先へ進んでいかないのではないかと。すなわち研究室のデータベースを共有することなどによって、専門分野の枠を飛び越えて研究することによって新しいものを生み出していこうということである。この先このようなインフラの整備がどんどん進んでいき、両者の融合が当然のことのように行われる日が来るあろうと私は確信している。こうした目に見える形での交流も重要ではあるが、さらに大切なのものは積極的に他の領域へ飛び込んで行こうとする人間の気持ちである。それが無ければ、こういった新たな動きも生まれてはこない。かくいう私も当初はかなりの戸惑いがあったことは否定できない。しかし工学系の人たちに会ってみて、研究分野は違えども根っこにある志は、まったく我々と変わらないことを強く認識した。現段階ではまったく何も成していない私であるが、この 1 年を通じて「研究を通して社会に役立ちたい」という思いを抱くきっかけとなったのが、こうした出会いであった。

第1章 人工市場の目的

私はこの論文のタイトルに**人工知能（Artificial Intelligence, AI）**という言葉を使用したわけであるが、これは人間が自分たちの頭と体を使うことなしに機械を介してどんなことがどこまでできるのかということをまず認識しておきたかったからである。単純に人工知能と聞いて思い浮かべるものは人間の形をしたロボットであろう。典型的なものとして二足歩行が可能であるロボットや、人間の言葉を認識するようなロボットが挙げられる。しかし今や人工知能という言葉は我々が想像しているよりも幅広い分野で使用され、また活用されているのである。

現在の人工知能研究には大きくわけて 2 つのスタンスがあるとされている(このように簡単に 2 つに分けてよいのかという意見もある)。1 つ目は人間の知能を兼ね備えた機械を作り出そうというスタンスで、俗に強い AI といわれている。これは人工知能と聞いて多くの人々がイメージするものであり、本当に知能がある機械のことを指す。2 つ目は、人間が知能を使ってすることを機械にさせようとするスタンス､これは逆に弱い AI といわれている。人工知能研究の多くは後者側に立っており、必ずしも人工知能＝ロボットという図式は成り立たない。様々な戦略を持つマシンエージェント(機械)を取引参加者として仮想市場に送り込み、人間の知的な行動のある部分を行わせている人工市場研究もこうした観点から見ると後者側にあてはまる。では我々の人工市場に参加するマシンエージェントは知的ではないのかというと一概にもそうとはいいきれない。我々が構築された市場システムの中でマシンエージェントに対してどのような情報を受け取らせ、そして自分は何をするべきなのかということをはっきりと明示してやることができるのであれば、エージェントプログラムは極めて優秀な性能を発揮する。そうして開発されたコンピュータプログラムは、少しは知的なものを持っていると言ってもよいであろう。将来的には人間の完璧な模倣を作り出すことも可能であるだろうが、それだけではなく人間がめったにすることがないふるまいであるとか、人間の計算能力を超えるコンピュータプログラムが実行できるのが知的なマシンエージェントの売りであるといえる。

しかし、人間のふるまいから大きく逸脱したエージェントばかりで構成された市場を作ったとしても、それではまったく現実世界との関連性を見出すことができない。ただ単に一風変わった市場を構築しただけになる。では人工市場研究の目的はどこにあるのであろうか。それはずばり人間心理を映し出す市場を作り出そうということである。大げさな言い方をすれば、市場という舞台の上に多種多様なエージェントを集めて、そこで繰り広げられる台本なしの即興劇を作ってみようということである。そこでは何が起こるかまったく予測はできない。ある人は他人のセリフに触発されて、それまで温めていた自分のセリフを変更するかもしれないし、また別の人は常識では考えられない突拍子もないことをやっ

てのけるかもしれない。この舞台はこれまでの伝統的な経済理論で考えられてきたすべての人間が同質で合理的なふるまいをするとする過度で理想的なものとはかなり異なる。伝統的金融理論ではそうした条件を仮定してきたため現実の経済と大きく乖離する非現実なものとなったと言っても過言ではない。要するにここで目指しているものは合理性から離れたものとして取り扱われていた人間の個性や心理的側面を重視する、よりリアリティな市場の構築なのである。

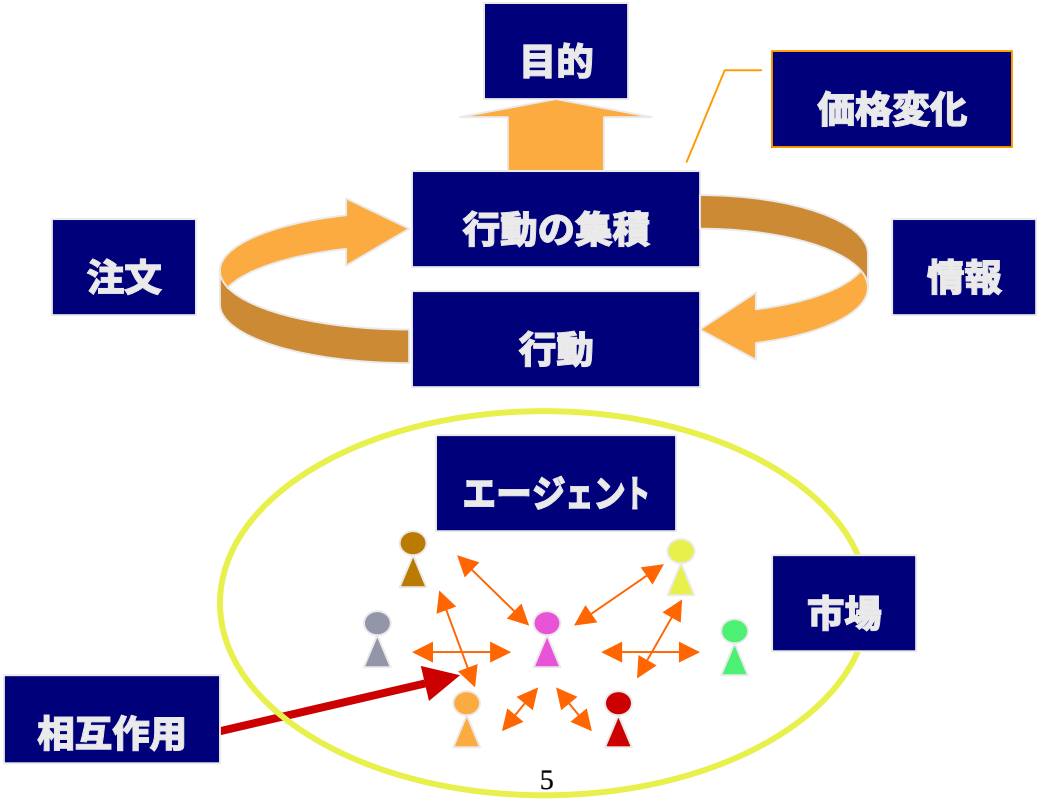
人間心理を映し出す現実的な市場を作るのであれば、何も機械にやらせることはせずに多くの人間を集めて実験を行った方が適当なのではないかという指摘があるが、そこにはまだいろいろと難しいことが多い。人間を介した実験では、被験者への参照情報が何であり、またそれをどのようにして扱っていくのかという問題があり、実験そのもののコンセプトが立てにくい。また実験を行う際には、何十人という人間を集めなくてはならなく、費用と時間がかかることが多い。そうした点で比較すると、人工市場は自分が希望する条件で何度も繰り返し同じ実験をすることが可能であり、また条件を簡単に操作することもできるという点で有利である。

次頁に人工市場の概要を示した。まず市場という枠組みの中に取引参加者であるエージェントが存在する。ここでの市場の枠組みの意味は、必ずしも１ヶ所にエージェントが集まって取引をするといったものではない。まず人工市場のエージェントは、市場内部で発生する先物価格などの市場情報を受け取る。取引が開始される初日にはエージェントが受け取る市場での内生データがまだ存在していないので、U-Mart では過去の現物データを価格情報として参照する仕組みになっている。そうした情報を受け取ったエージェントは、自分自身が持っている取引戦略に受け取った情報を照らし合わせて、市場での行動を決定する。ここでいう具体的なエージェントの行動とは、株式を売ろうか、それとも買おうかという「売買決定」、それではいかほどの価格でいかほど量を売買しようかという「注文価格・注文量の決定」などが挙げられる。様々な取引戦略を持つエージェントのこうした行動が集積することによって、人工市場では独自の市場現象が発生することになる。市場現象にはもちろん価格変化も含まれる。そうした価格変化などの市場の動向は、新たな情報としてエージェントの元に舞い戻ってくることになる。さらに今度はその情報がエージェント自身の行動を決める大きな要因となって行動が繰り返されるというわけである。

エージェントの行動決定において、もうひとつ重要な要因として考えられているのが、それぞれのエージェント間に潜む相互関係である。これはどういうことであろうか。単純ではあるがこのような事例があったとしよう。A という人物がある日の午後に市場で C という銘柄の大規模な買い注文を行うらしいという情報を B という人物が事前に掴んだと仮定する。その情報が手に入るまで B は C 銘柄を午前のうちに売却する予定であった。しか

し、その情報によって B はそれまで予定していた行動を変更して、A の大量の買いよって午後に C 銘柄の株価が上昇するまで売却を待つという戦略を採ることにした。午後になって、情報通り A は大量の買いを入れ、C 銘柄の価格は上昇し、B はそこで売り抜けることに成功した。ここでは明らかに B の行動は A の売買行動の影響を受けており、売買判断を決定する際の主因となっている。さらに今度は別の視点から考えてみよう。A は自身の売買情報が B へと漏れていることを予測して、B の売買行動を先読みし、売買判断を決定する。ここでは A 自身への売買決定にも影響を与えることになるのである。この読み合いは永久に繰り返すこともできる。また 2 人だけではなく複数の人物も含めて永久に続けることもできるのである。取引参加者(エージェント)間で発生するこうした影響の与え合いは、**相互作用（interaction）**といわれるものである。相互作用が意味するところは、ある個人の行動が他の人々の行動決定にも何らかの影響を与えているということである。

こうしたエージェント間に関係する相互作用から前述したエージェントの行動が集積することで生まれる市場現象が現れてくるという枠組みの中で、マクロとミクロに潜むメカニズムを見つけだそうというのが人工市場研究の目的だと考えられる。これまでの経済学の研究の多くは、仮説の検証をすることが大きな目的のひとつであった。もちろんこの研究でも人工市場というシミュレータを使用して既存の理論を説明しようとすることは可能であるし、またそうした研究もなされている。しかし人工市場研究は、それだけにとどまらずに、そのシミュレータを使ってマクロ-ミクロ間の新しい法則性やパターンを発見しようという部分に大きな特性があると思われる。この目的が成されたときにこそ、私がこの研究を始めたときに思った「個人の持つ特性や心の動きが市場に対してどのように反映されているのか知りたい」という願望がかなえられるのかもしれない。

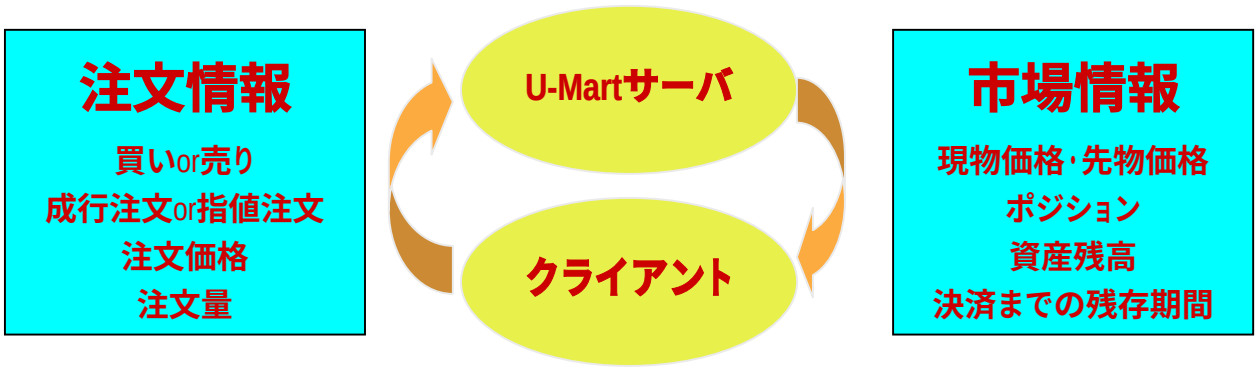


第2章　U-Mart (Unreal Market as Artificial Research Test-bed)

U-Mart は基本構造が Java で書かれている人工市場システムで、そのすべてを日本の研究者チームが開発している。研究者チームには工学系の研究者だけではなく、社会学、経済学、心理学の研究者も参加している。U-Mart では取引を行うマシンエージェントを公募し、公開実験を行うイベントを年 1 回開いており(U-Mart2002)、これまでに 10 以上の大学、大学院、企業が自ら開発したプログラムを持ち寄って、マシンエージェントが稼ぎ出す利益を競うという形式で、研究成果を発表している。ここで私はエージェントの前にあえて「マシン」という言葉をつけたわけであるが、U-Mart では取引戦略をコンピュータプログラムされたマシンエージェント同士の取引だけではなく、人間も専用のインターフェイスを使い、ヒューマンエージェントとして取引に参加することができる。もちろんヒューマンエージェントとマシンエージェントが混在した市場を構成することも可能である。情けないことではあるが、自分が開発したマシンエージェントに開発者が負けることもよくある話である。

図 1　ヒューマンエージェント専用インターフェイス

単純に U-Mart を構成しているものを考えると、人とマシンが混在する**取引クライアント**とインターネット上にある **U-Mart サーバ(仮想取引所)**である。この U-Mart サーバがそれぞれのクライアントへと市場情報を出力し、クライアントは注文を仮想取引所へ入力するという仕組みとなっている。インターネット上に仮想取引所が置かれると書いたが、ネットワークを通じて遠隔者との取引をできるのも U-Mart の特徴のひとつある。東京にいる人間と札幌にいる人間が同時に同じ市場で取引することも可能なのである。



これまでは簡単な U-Mart の特性を書いてきたが、具体的に U-Mart では何を取引しているのであろうか。U-Mart では実在する株価指数を元にして、仮想的な先物取引ができるようにつくられている。U-Mart で取引されるのは先物指数だけであり、現物指数の取引はできない。その現物指標として使用しているのが実在する株価指数 **J30** である。J30 は毎日新聞が日本を代表する企業群 30 社を選定し、日本経済の的確な動向を示す株価の指標として開発したものである。J30 はバブル崩壊直前の 1989 年 12 月 29 日の株価をスタート基準とし、30 社の株価を平均している。U-Mart には数年分の J30 の価格系列がデータとして組み込まれているのである。U-Mart では価格決定方式に板寄せ方式を採用している。これはどういう方式であるかというと、市場全体の需要と供給をいったん全て集めて需給が釣り合うところに価格(レート)を決めるやり方である。こうした方式で形成された先物価格は、独自の架空のものであるので、現実の価格に影響を与えない。つまり U-Mart では、参加者ひとりひとりの売り買い（行動）が先物市場に直接影響を与えることになる。野村證券が行っているような一般的なバーチャル取引とは一味違うということになる。相当な高値で買いの注文を出すと当然の如く先物市場は激しく動くし、またその逆も同じである。身をもって市場の動きを感じることができる仮想先物市場シミュレータなのである。

第3章 マシンエージェントの取引戦略

それでは具体的に U-Mart に参加しているエージェントはどのような戦略を武装しているのであろうか。U-Mart ではそれぞれのエージェントが異なった行動をとるところに特徴があるということは先述した通りである。その中でも人間の思考や心理を組み込んだエージェントを開発することは、この研究における我々のひとつの重要な目標である。しかし、我々がこれまでに開発してきたエージェントは、既存のテクニカル分析理論を元に改変したものばかりである。これは私がオリジナルの取引戦略理論を構築ができなかったことが主因であるが、プログラミング能力がまだまだ未熟であることも要因のひとつである。だが、プログラミング初心者がテクニカル分析の理論を元にしてエージェントを作成することはこの分野の導入学習としては非常に効果的であると感じた。それは U-Mart が取引参加者に提供しくれる情報がテクニカル分析向きであるので、そうした情報を受け取って売買判断を下していくというエージェントが比較的容易に作成できること、また理論的にはそれほど難解ではないものが多く、コード化しやすいことがあげられる。この章では我々が 1 年間かけて習得していったプログラミング学習の過程を追いながら、我々が開発したエージェントと U-Mart 用エージェント開発キットである UMIE2002SDK(Software Development Kits)に導入されている基本的なエージェント戦略を紹介する。

そもそも私がエージェントを開発しようと思いたったのは、2002 年 6 月にアメリカのカーネギーメロン大学（CMU）で開かれた CASOS（Computational and Mathematical Organization Theory Conference）で U-Mart チームが国際実験 UMIE2002（U-Mart International Experiment 2002）を行ったことに端を発する。国際実験をするにあたって U-Mart チームは、様々な取引戦術をもつエージェントを募集し、それに我々が挑戦してみようという形で、エージェント開発をすることになったというわけである。エージェントを作成するには、個性ある取引戦略を開発することも重要であるが、まず初めにエージェント戦略を書くためのプログラミング言語（Java）を学習しなくてはならなかった。これまで、まったくプログラミング言語を学習したことがなかった我々は、基礎的な部分を大雑把に教えてもらおうと、防衛大学校の佐藤浩先生に Java の基本コードと U-Mart の仕組みを 5 月の頭に教授していただいた。ここで教わったことは本当に Java の基礎の基礎であるが、これが今でも私の Java 言語能力の根幹を成しているといっても過言ではない。それではここでエージェント開発を行う際に必要な Java の概念をまとめておこう。

まず他の人たちが設計したさまざまなクラスを利用するようなときには、同じ名前のクラスを混在させて扱わなくてはならない場合がある。このときに使うものが、**パッケージ（package）**という概念である。パッケージは、複数のクラスやインターフェイス（この UMIE2002SDK では出てこないが）をまとめ、名前の重複を避けるために使われる。

- 構文　＜**package パッケージ名；**＞

我々のファイルでは、TestStrategy クラスがパッケージ strategy に含まれるということになる。コンパイルの際に　javac **strategy**/TestStrategy.java と書くが、このときの **strategy** はパッケージ名を指している。ここで注意しないといけないのは、package を指定しないと、そのクラスはすべて「名前のないパッケージ」の中に含まれるということだ。

Java には、よく使われる機能をまとめたクラスライブラリ（class library）と呼ばれるクラスの集まりが用意されている。java.util パッケージはそのひとつであり、また代表的なクラスライブラリでもある。このパッケージのユーティリティクラスには、プログラミングをする上で使用頻度の高い基本的な機能が含まれている。たとえば、日付や時刻の情報を取り扱ったり、乱数を生成したり、カレンダーを利用したりすることができる機能である。またパッケージ名を毎回記述していたのでは、非常に手間がかるので、ファイルの先頭に、次のような指定をすることができる。

- 構文　＜**import パッケージ名．クラス名**＞

この記述をしておけば、コード内でクラスを記述する場合、パッケージ名を記述することなく、クラス名だけを記述してもよいということになっている。これは**インポート（import）**といわれるものである。ようするに、クラス名だけで異なるパッケージのクラスを記述できるようになるということである。また、クラスライブラリのクラスを利用するには、パッケージ名を指定しなくていいようにするために、ファイルの先頭でインポートするのが普通である。java.util.の後に＊がついているのは、同じパッケージ内のすべてのクラスを取り込むということを意味している。これはいちいち使うクラスを指定しなくてはならないという作業を省略するためである。

我々が最初に記述したプログラミングでは、すでに作成してある「Strategy クラス」をもとにして、我々の独自の戦略である「TestStrategy クラス」を効率よく作成していった。このように新しいクラスを作成することを、**クラスを拡張する（extends）**という。新たなクラスは、既存のクラスのメンバを「引き継ぐ」ようなかたちとなる。既存のクラスに新しく必要な性質や機能（メンバ）をつけたすように、我々の取引戦略コードを書くことができるのである。TestStrategy クラスは Strategy クラスのメンバを受け継ぐので、Strategy クラスで既に記述したメンバについて、もう一度記述する必要はない。TestStrategy に独自のデータと機能だけをまとめて記述すればよいのである。このように、新しく拡張したクラスがメンバを受け継ぐことを、**継承（inheritance）**と呼ぶ。このとき、もとになる既存のクラスは**スーパークラス（superclass）**、新しいクラスは**サブクラス（subclass）**と呼

ばれる。ここでは「Strategy クラス」がスーパークラス、そして「TestStrategy クラス」がサブクラスとなっている。

Math クラスは数学的な演算を行う機能をまとめたクラスである。我々が UMIE に提出したファイルでは次の３つのクラスメソッドを使用している。

Math max（int i, int j）　ｉとｊの大きい方の値を返す
Math min（int i, int j）　ｉとｊの大きい方の値を返す
Math abs （int i）　　ｉの絶対値を返す

この段階のことまでを理解した上で我々が作成したのが以下の 4 体のエージェントである。

① 裁定取引エージェント
② ウィリアムズ％Ｒ＋裁定取引エージェント
③ 同上（エージェントが持つ内生データが異なる）
④ ウィリアムズ％Ｒ＋裁定取引＋移動平均エージェント

UMIE2002SDK の GUI サーバ上で取引を行う場合、裁定取引を組み入れた戦略は絶大な効果を発揮する。GUI サーバにある 20 体のサンプルエージェントのうち、裁定取引を用いているのは SFspread Strategy の 2 体のみである。このため、現物価格と先物価格は何日ものあいだ大きく乖離することがある。しかしながら、今回の国際実験では多くのエージェントが裁定取引を組み入れてくることが予想されたため、乖離している時間は圧倒的に短くなり、価格差は小さくなる。先物価格が現物価格よりも高いときに多くのエージェントが売り注文を出すと、先物価格は下落して現物価格との差を縮めてしまう。逆に、先物価格が現物価格よりも低ければ、多くのエージェントが買い注文を出すため、先物価格は上昇し価格差は縮まってしまう。裁定取引は現物価格と先物価格が長時間大きく乖離していることにより効果を発揮する戦略である。つまり、現物価格と先物価格が乖離していないときには利益を生み出しにくくなる。したがって、純粋な裁定取引のみの戦略は今回の国際実験おいて通用しないと推測した。裁定取引が効果を発揮できない以上、他の戦略の採用を考えなければならない。そこで我々が興味を持ったのはオシレータ系のテクニカル分析であるウィリアムズ%R である。ウィリアムズ％Ｒは、アメリカのシステム・トレーダーであるラリー・ウィリアムズが考案したものである。他のオシレータ系の分析手法と同様、「エッジ」と呼ばれる一定の値を設定する。そして、％Ｒがエッジを超えたときに売買を行う。ウィリアムズ%R は、５日間の高値と終値の差を高値と安値の差で割った値である。

計算式

$$\%R = (Hn - C) / (Hn - Ln) \times 100$$

Hn = 期間 n における最高値
Ln = 期間 n における最安値
C = 当日終値
＊我々の戦略では今日の終値は直近の先物価格を採用した

売買の判断は次の様にして決定される。%R がエッジよりも低いときには買い。%R が（１－エッジ）よりも低いときは売り。ウィリアムズ％Ｒを用いるとき、一般的にエッジは 20%に設定されるが、数十回にもわたってエッジを変えてシミュレーションをした結果、今実験では 17%を採用することにした。また、高値と安値の参照期間は 10 期間とした。

①の「裁定取引エージェント」は純粋な裁定取引をする。先物価格と現物価格が少しでも乖離している場合に取引をする。現物価格と先物価格が「ａ%以上乖離しているときにのみ取引をする」というような条件はつけなかった。これは取引量を増やすためである。指値は前期の先物価格に最大 60 円の誤差（20×平均 0、標準偏差１の正規分布）を加えたものを採用し、取引量は 50 単位に最大 5 単位の誤差を加えたものを採用した。乱数をもちいた理由は、シミュレーションで良い成績を残したことに起因する。②と③の「ウィリアムズ％Ｒ＋裁定取引エージェント」は①のエージェントの裁定取引の条件を満たし、かつ、前述したウィリアムズ%R の条件を満たした場合にのみ取引をする。②のエージェントは指値に関して①のエージェントと同じである。取引量に関しては、60 単位に最大 10 単位の誤差を加えて取引をする。③のエージェントは指値の誤差を先物価格と現物価格の差に乱数を掛けたものにしている。それ以外は②のエージェントと同様である。④のエージェントは確率 2/7 でウィリアムズ％Ｒ戦略を、確率３/７で裁定取引戦略を、確率２/７で現物移動平均戦略を用いるエージェントである。現物移動平均戦略は 10 日間の現物価格の移動平均よりも現物価格が高いときには売り注文を出し、低いときには買い注文を出す。現物移動平均戦略はウィリアムズ％Ｒが利益を出し難いチャートの動きのときに強い戦略であることをシミュレーション中に感じたので採用した。指値と取引量は①のエージェントと同じである。なお、マックス・ポジションは 4 体のエージェントとも 300 単位である。

これらの 4 体のエージェントは我々が行ったスタンドアローン型のシミュレーション実験（ここで登場するサンプルエージェントの戦略はこの後に記述する）では損失をほとんど出さなかった。これは複数の取引戦略をエージェントに組み込むことによって、注文を

出すときの条件をより厳しく設定したからと考える。こうすることでより好条件で売り買いすることが可能になり、損失をほとんど発生させずにすんだ。しかし、売買条件を厳しくすることは、同時に売買機会を減少させることにもつながった。単一の取引戦略を持つエージェントと複数戦略を持つエージェントの取引量を比較すると、6 割から 7 割も複数戦略エージェントの取引量が減少することがわかった。さらに 3 つ 4 つと取引戦略を付け合せていくと、まったく取引を行わないという現象も発生した。また取引量が大幅に減少することによって、安定的に僅かに利益は出すものの大勝ちをするということがまったくなくなった。だが、スタントアローン型のシミュレーションにおいて、サンプルエージェントが大勝ちするのには次のような理由があると私は考えている。他のエージェントを引き離して利益を大きく獲得する状況は、安定的な売買注文を出しながらそのエージェントが持つ戦略が採用する現物価格データにしっかりはまったときに起こりやすい。実際に、ランダムに注文を出していくエージェントがこの実験で大勝ちすることが多いのもその類であろう。つまりこのシミュレーション上での大勝ちは、この先価格が上昇しそうだから大量に買い注文を出しておこうといったような一種の賭けのような要素を含んでいる。サンプルエージェントは大勝ちと同程度に大負けも多く起こっているのである。こうした状況の中でも常に大勝ちをするエージェントを開発することは我々の理想ではあったが、期日までにはそこまでものを開発するまでには至らなかった。こうして我々のエージェントには、現実の多くの投資家が求める「ほとんど損失を出さずに小額でも確実に利益を出す」という取引戦略を持たせた。安定的な利益を生み出すという点でこれらのエージェントに一定の評価を下したわけであるが、憂慮する面も多々あった。それはこのスタンドアローン実験に参加させたサンプルエージェントがすべて単一の単純な取引戦略しか持っておらず、また裁定取引戦略には弱いということである。つまりこの個人的な実験では安定的に利益を出すことができたが、この結果が強いエージェントを開発したということにはつながらないということである。より複雑な取引戦略を持つエージェントと取引を行った場合には、今回の実験結果とは大きく異なることもありうる。実際、我々のエージェントを他の人々が作成したエージェントと取引をさせたところ、利益の順位を大幅に下落させることになった。

UMIE2002 に参加させた 4 体のエージェントは、正直言って「とりあえず作ってみる」という感が色濃く出ていたところが否めなかった。安定的な利益を出し続けることだけを重視し、エージェントの個性が埋没してしまったような気がする。そこで 2002 年 11 月に大阪市立大学で開催された U-Mart2002 に参加するにあたって作成したのが、サイコロジカル・ラインを用いたエージェントである。ここでもテクニカル分析の域を出ることができなかったわけであるが、その中でも可能な限りより人間の心を反映させるようなものを作りたいという思いから作成したものである。

サイコロジカル・ライン（Psychological Line）は、株価変動を心理の見地から捉えたもので、マーケットの心理、すなわち市場参加者の心理を測って売買の判断を行うオシレータである。その基本コンセプトは、「相場は、上昇も下降もいつまでも続くものではない」という特性を「心理的な」ものとして捉え、一定期間内で相場が上昇した日数が極端に多いと上げ過ぎ、下落した日数が多ければ下げすぎと見なし、それを指数化して見ることを目的にしたオシレータある。

一般的に、より売買のポイントを掴むには「サイコロジカル・ライン」より「値動きサイコロジカル・ライン」の方がすぐれていると言われている。私が U-Mart2002 で採用したのはこの戦略である。「値動きサイコロジカル・ライン」は日柄では無く、上昇幅・下落幅を考慮し、「値動きに対する心理的なもの」を数値的に見るオシレータである。

計算式

値動きサイコロジカル・ライン（PL）＝（U÷（U＋D））×100

PL = (DownPriceSum) / (DownPriceSum + UpPriceSum);

U（UpPriceSum）　＝N 日間の前日比上昇幅の合計

D（DownPriceSum）＝N 日間の前日比下落幅の合計

つまり、計算期間中の前日比がプラスであるものの合計を、計算期間中の前日比の絶対値の合計で割ったものとなる。ちなみに、私が作成したエージェントは参照期間 14 日を取っている。数値の見方としては、75%以上が買われ過ぎ、25%以下が売られ過ぎ、50%が中立となり、売買のタイミングは次の通りである

買いシグナル（売られすぎているとき）

1. サイコロジカル・ラインが 25%以下で反転したとき。
2. サイコロジカル・ラインが 25%以下で反転し、その後 25%以上になったとき。

売りシグナル（買われすぎているとき）

1. サイコロジカル・ラインが 75%以上で反転したとき。
2. サイコロジカル・ラインが 75%以上で反転し、その後 75%以下になったとき。

次頁に示したのは私が作成したこのエージェントのソースコードである。

```
package strategy;
import java.util.*;
```

```
/**
  *Fluctal Psychological Line (Psychological Strategy)
  *
  *@author Shigeto Kobayashi (Chuo University)
  */

public class TestStrategy extends Strategy {
  private Random random;
  private final int widthOfprice = 100;
  private final int maxQuant = 60;
  private final int minQuant = 10;
  private final int maxPosition = 300;
  private final int nominalPrice = 3000;
  private final double edge = 0.2;

  public TestStrategy( int seed ) {
    random = new Random(seed);
}
    public Order getOrder(int[] spotPrices, int[] futurePrices, int pos, long money, int
restDay) {

    Order order = new Order();
   int latestSpotPrice = spotPrices[119];
   int latestFuturePrice = futurePrices[59];

    if ((latestSpotPrice <= 0) || (latestFuturePrice < 0)) {
      order.buysell = Order.NONE;
      return order;
    }
    int UpPriceSum = 0;
    int DownPriceSum = 0;

    for (int i=0; i < 12; i++) {
      if ((futurePrices[i+47] - futurePrices[i+48]) > 0) {
        UpPriceSum += Math.abs(futurePrices[i+47] - futurePrices[i+48]);
      } else {
```

```
            DownPriceSum += Math.abs(futurePrices[i+47] - futurePrices[i+48]);
          }
        }
        double PL = (double)(DownPriceSum) / (double)(DownPriceSum + UpPriceSum);

        if (PL < edge) {
          order.buysell = Order.BUY;
        } else if (PL > 1.0 - edge) {
          order.buysell = Order.SELL;
        } else {
            order.buysell = Order.NONE;
          return order;
        }

        if (order.buysell == Order.BUY) {
          if (pos > maxPosition) {
            order.buysell = Order.NONE;
            return order;
          }
        } else if (order.buysell == Order.SELL) {
          if (pos < - maxPosition) {
            order.buysell = Order.NONE;
            return order;
          }
        }

        while ( true ) {
          order.price = latestFuturePrice + (int)(widthOfprice * random.nextGaussian());
          if (order.price > 0)
          break;
          }
        order.quant = minQuant + random.nextInt(maxQuant - minQuant + 1);
        return order;
      }
    }
```

結果から先に述べるとこのエージェントはそれほど利益を上げることはできなかった。先述のサンプルエージェントたちとの取引でも損益がプラスになるのは五分五分であった。しかし、このころから私が抱き始めていた「エージェント開発の意味」に少しだけ答えを与えてくれるきっかけになった一体にはなった。

これまでの我々の方針は安定的な利益を出し続けるエージェントの作成であった。そのために我々が重視していたのは、確固たる理論に基づく取引戦略であった。つまりある程度、複雑な理論をコードとして記述さえできれば、単純な取引戦略しか持たないサンプルエージェントには楽勝できると信じていた。だが結果はその通りにはならなかった。何故であろうか。単純にその戦略が弱かったと考えることもできるが、サイコロジカル・ラインは実際の取引にも活用されている戦略であるので、そのことだけで結論付けるのも尚早ではなかろうか。私が予測したのは、エージェント同士の組み合わせの問題である。それぞれのエージェントが持つ取引戦略にも、人と人の間に存在する相性のようなものがあるのではなかろうかということである。しかし、この問題の裏付けはまだ取れてはいない。エージェント間に存在するように見えた相性は、実はエージェントが市場のダイナミックな動きに自身の戦略を適応しきれなかったことによるものかもしれない。このことに焦点を当ててエージェント作りをしてみようというインセンティブが湧いてきている。エージェント間の相互作用を意識したときに、他のエージェントのふるまいを刺激する行動とはどのように形成されるのであろうか。これからのエージェント開発のひとつのテーマとなった。

次に UMIE2002SDK に入っているサンプルエージェントの取引戦略を解説する。サンプルエージェントの取引戦略は大きく 9 つに分類することが出来る。どの取引戦略も理論は単純で理解しやすいものとなっている。まず注文量はどのエージェントも同じ方法で決定しており、計算式は

注文量＝ minQuant + 乱数×(maxQuant - minQuant + 1)

minQuant = 50　maxQuant = 15

任意の定数 minQuant と maxQuant の量を変えることによって注文量を調整する

１．SFspreadStrategy　　裁定取引戦略

spreadRetio　＝（前回の先物価格－前回の現物価格）/（前回の現物価格）
spreadRetio　≦　－0.01　⇒　買い
spreadRetio　≧　　0.01　⇒　売り
その他の場合　　⇒　取引しない

mine price　＝（前回の先物価格＋前回の現物価格）/ 2
sigma　＝ |(前回の先物価格－前回の現物価格)| / 2

注文価格　＝　mine price ＋　sigma ×　乱数

２．SRandomStrategy　　現物ランダム戦略

売買をランダムで決定する　０が出たら　⇒　取引しない
１が出たら　⇒　売り
２が出たら　⇒　買い

注文価格　＝　前回の**現物**価格　＋　widthOfPrice　×　乱数
widthOfPrice　＝　20

３．RandomStrategy　　先物ランダム戦略

売買をランダムで決定する　０が出たら　⇒　取引しない
１が出たら　⇒　売り
２が出たら　⇒　買い

注文価格　＝　前回の**先物**価格　＋　widthOfPrice　×　乱数
widthOfPrice　＝　20

４．TrendStrategy　　トレンド戦略

前回の先物価格　＞　前々回の先物価格　（価格上昇トレンド）　⇒　買い
前回の先物価格　＜　前々回の先物価格　（価格下落トレンド）　⇒　売り
その他の場合　⇒　取引しない

注文価格はランダム戦略と同様

５．AntiTrendStrategy　　アンチ・トレンド戦略

前回の先物価格　＜　前々回の先物価格　（価格下落トレンド）　⇒　買い
前回の先物価格　＞　前々回の先物価格　（価格上昇トレンド）　⇒　売り
その他の場合　⇒　取引しない

注文価格はランダム戦略と同様

６．SmovingAverageStrategy　　現物移動平均戦略

移動平均
＝　参照期間（この場合は取引 10 回分）における現物価格の合計値　／　取引回数
前回の現物価格　＞　移動平均　⇒　買い
前回の現物価格　＜　移動平均　⇒　売り
その他の場合　⇒　取引しない

注文価格はランダム戦略と同様

７．movingAverageStrategy　　先物移動平均戦略

移動平均
＝　参照期間（この場合は取引 10 回分）における先物価格の合計値　／　取引回数
前回の先物価格　＞　移動平均　⇒　買い
前回の先物価格　＜　移動平均　⇒　売り
その他の場合　⇒　取引しない

注文価格はランダム戦略と同様

８．DaytradeStrategy　　デイ・トレード戦略

買いと売りの注文を毎回同時に行う

買い注文価格　＝　前回の先物価格　×　（ 1.0 － spreadRatio ）
spreadRatio　＝　0.01
売り注文価格　＝　前回の先物価格　×　（ 1.0 ＋ spreadRatio ）
spreadRatio　＝　0.01

９．RsiStrategy　　相対力指数（RSI）戦略

RSI（相対力指数）はワイルダー（J.Welles Wilder,Jr）が株式、商品、株式指数などのバー・チャートと共に使用するために開発したものである。RSI は、ある銘柄の過去のパフォーマンスと現在の価格を比較する指標である。RSI の値は 0 から 100 の範囲で変化する。

RSI が 70 から 80 の場合には、その銘柄は買われ過ぎであり、買いを控えるシグナルとなる。また、RSI が 30 から 20 の場合には売られ過ぎであり、売りを控えるシグナルとなる。UMIE2002SDK キットに入っている RsiStrategy では、RSI が 70 以上で売り、30 以下で買いとなる。その他は取引をしない。

RSI の計算

RSI では、平均価格の割合を算出し、それを 0 から 100 までの数値に標準化する。

$$\text{RSI（相対力指数）} = 100 - \frac{100}{\text{RS}+1} = 100 - \frac{100}{U/D+1} = \frac{U}{U+D} \times 100$$

$$\text{RS（相対強度）} = \frac{\text{N日間で先物が上昇した日の上昇額の合計額（}U\text{）}}{\text{N日間で先物が下落した日の下落額の合計額（}D\text{）}}$$

株価が上昇した日の上昇額とは、前日より終値が上昇した日の変化額であり、株価が下落した日の下落した日の下落額とは、終値が下落した日の変化額である。

ワイルダーはもともと、計測日数 N を 14 日としていたが、その他によく使われるのは 9 日と 21 日である。UMIE2002SDK では 10 日を採用している。

次に紹介するのは私のサイコロジカル・ラインエージェントと共に U-Mart2002 にチームメンバーとして参加したエージェントである。U-Mart2002 にでは上位に入賞することができなかったが、理論を忠実に再現しているエージェントであるといえる。私が次章で行った実験にも参加させている。作成はゼミの後輩である石山洸くん、中田翔くん、原田圭くんが行った。以下は中田くんが執筆したエージェント解説を一部引用し、要約したものである。

ストキャスティクスはオシレータ系の指標のひとつである。ストキャスティクスでは K ライン（%K）、D ライン（%D）、SD ライン（Slow%D）と呼ばれる 3 種類のラインが用いられるが、売買判断をする際に使用するのは 2 本のラインである。

今回のエージェントの売買判断には K ライン、D ラインの 2 本のラインを用いた**ファースト・ストキャスティクス（FastStochastics）**を採用した。ファースト・ストキャスティクスも基本的には他のストキャスティクス同様、0%に近いほど売られ過ぎ、100%に近いほど買われ過ぎを示している。

計算式

$$\%K = (C - Ln) \div (Hn - Ln) \times 100$$

Hn = 期間 n における最高値

Ln = 期間 n における最安値

C = 当日終値

ストキャスティクス%K は、一定期間（n日）の最高値から最安値の間で、当日終値からその最安値を引いた幅が、何%を占めるのかを百分率（0%～100%）の指数で表したものである。0%に近いほど売られ過ぎ、100%に近いほど買われ過ぎということになる。

一般的に期間は５日と設定、ストキャスティクスの 70%以上を買われ過ぎ、30%以下は売られ過ぎとし、30%以下で買い・70%以上を売るのが一般的である。

計算式

$$\%D = (C - Ln\text{の}m\text{日間の合計}) \div (Hn - Ln\text{の}m\text{日間の合計}) \times 100$$

Hn = 期間 n における最高値

Ln = 期間 n における最安値

C = 当日終値

ストキャスティクス%D は、ストキャスティクス%K をm日ゆるやかにしたものである。一般的な期間の設定は、「n＝ 5、m＝3」で、ストキャスティクスの 70%以上を買われ過ぎ、30%以下は売られ過ぎとし、30%以下で買い・70%以上を売るのが一般的である。

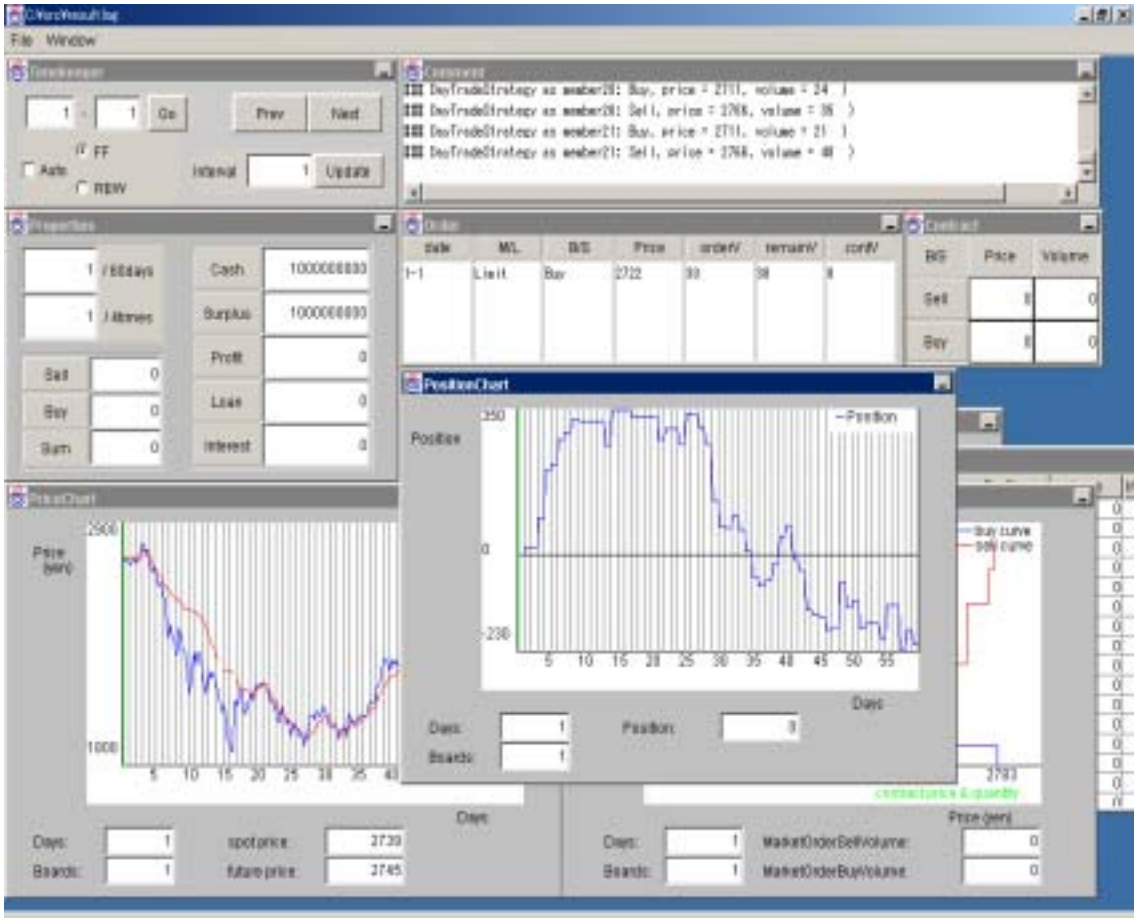
売買の基準は２本のラインのクロスで、K ラインが D ラインを下から上に抜けてきた時に買い、上から下に抜けてきた時に売りとする。しかし、何度もクロスが発生するので、ダマシが多くなる。このため、「30%以下の位置で上に抜けたら買い」、「70%以上の位置で下に抜けたら売り」というフィルターをかけている。

第4章 実験のためのプログラム改良

平均株価が現在よりも遥かに高かったバブルの頃は、総じて株式の売買高も多く、逆に景気が低迷している現在のように平均株価が低いときには株式の売買高が少ないと言われる。本当に株価の変動と売買高には何らかの関係があるのであろうか。これまでの実証研究では２者間にはおおむね正の関係があるとされており、株価が下落傾向にあるときよりも上昇傾向にあるときの売買高の方が大きいとされている。それでは超短期市場である U-Mart ではどのような動きをするのであろうか。これまでの U-Mart では実験環境を整え、そしてマシンエージェントを持ち寄って利益を争うという研究は多く行われてきたが、人

工市場でおこる市場現象の詳細な研究がそれほど行われてこなかった。まず導入部分として U-Mart における株価変動と売買高の関係をシミュレートしようと考えた。より人間臭い市場を構築するというスタンスでここまで望んできたのだが、この実験ではバブル期のような社会的背景を再現することであるとか、エージェントに複雑な心理的要因を組み込むことができなかった。これは私の技術面の未熟さが大きな原因であるが、実験を行うのが初めてということなので、よりシンプルな条件での市場の動きを検証したかったという私の願望も入っている。また、実験に参加するマシンエージェントの構成を変えることによってエージェント間潜む相互関係が結果にどのような影響を与えるのかにもスポットをあてることにした。

図２　U-Mart スタンドアローン版シミュレータ用ログビュア



実験を行うにあたって、まず実験環境を整えることにした。前頁に示したのが U-Mart のスタンドアローン版シミュレータ用のログビュアである。このように、U-Mart に参加しているマシンエージェントの取引結果や取引全体の流れは GUI で現れるようになっている。ただ、このログビュアは視覚的には大変見やすいが、複数回の実験を行って、比較分析をするのには不向きである。そこで大量にはき出されるログを自動的に CSV 形式で保存し、

計算を行ってくれる実験ツールを開発することにした。そうすることによって自分たちが欲しい情報だけを確実に取り込め、エクセル上で計算し易くすることができる。開発はすべて Java で行い、ゼミの後輩である石山洸くんがその大部分を担ってくれた。今回の実験では石山くんが開発したそのプログラムツールに私が加筆して修正を加えたものを使用した。

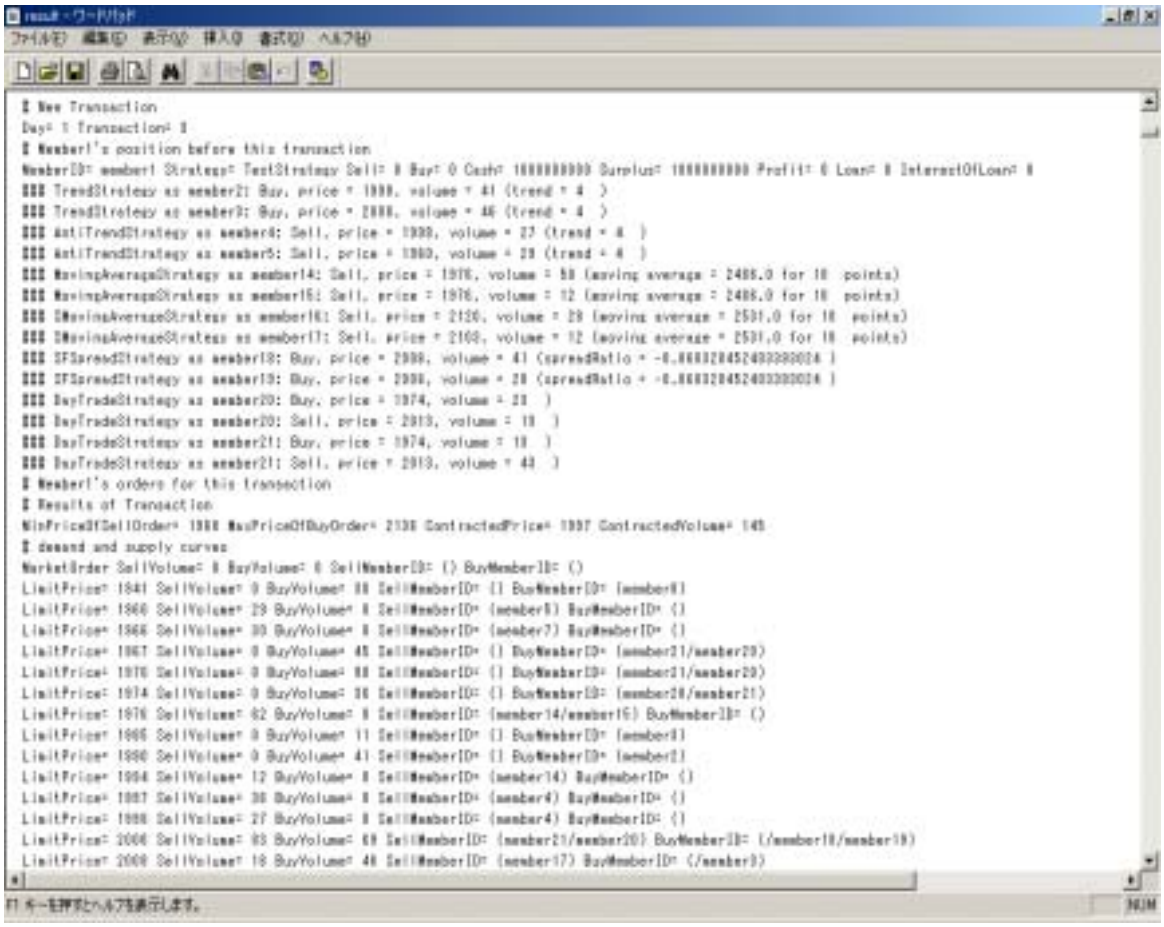


図 3　U－Mart スタンドアローン版シミュレータ用取引ログ

U-Mart で行われた取引ログは上図のような形で表れる（図 3)。これは 1 日 4 回ある板寄せのうちの 1 回分である。U-Mart は基本的には 60 日で取引を終える仕組み（板寄せは 1 日 4 回）なので、1 回の実験でこの 240 倍の情報が一気にはきだされることになる。この中から必要な情報を取り出すのは並大抵の労力ではない。そこで開発されたのが分析ツールである。それを使用することによって、ログを見ながら自分が必要とする情報をいちいち見ることなく、自動的に抽出して計算してくれることが可能となった。今回のログファイルから取り出したのは、各板寄せ毎の「現物価格」「先物価格」「約定数量」である。
結果はエクセル上に次頁の図のような形式で自動的に現れる（図 4)。左から「板寄せ期間」「現物価格（SpotPrice)」「先物価格（FuturePrice)」「約定数量（ContractedVolume)」

を表示する。こうした形式で情報を引き出すことができるので、先物価格と現物価格の推移を表したグラフなどが簡単に描くことができる。その他にも応用的な使い方がまだまだ出てくるであろう。

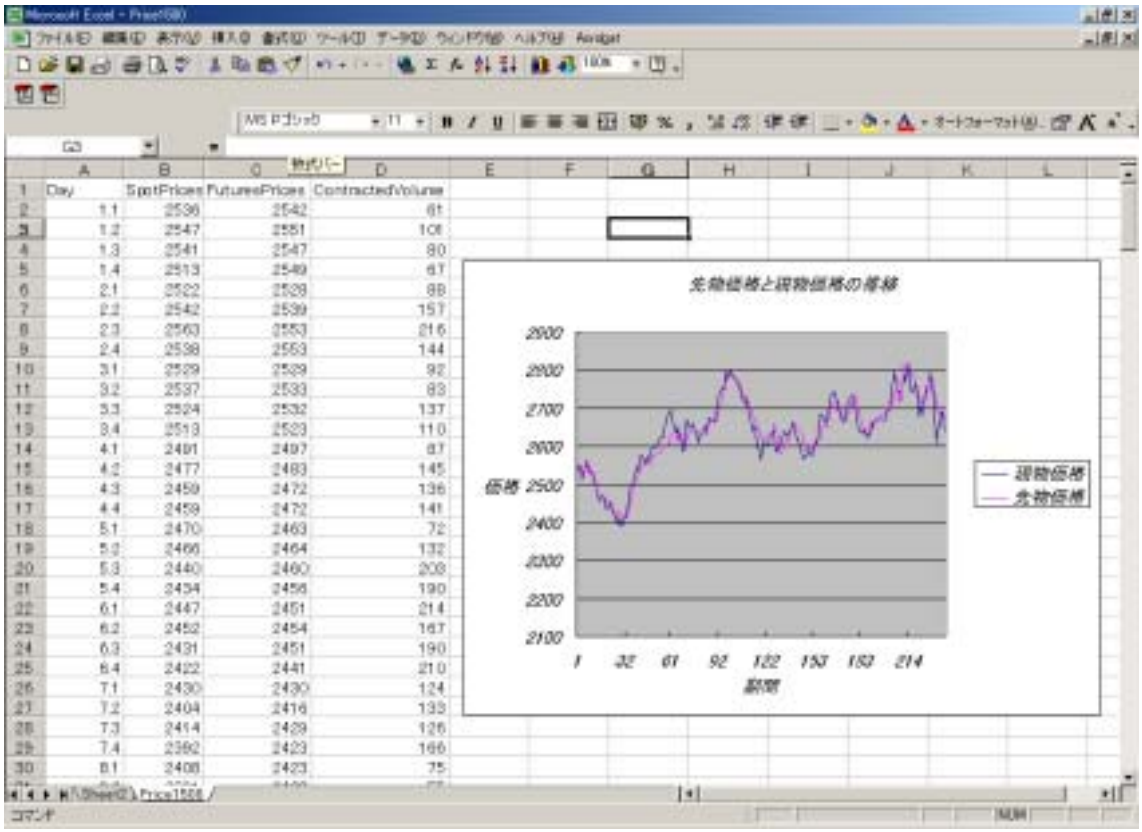


図 4　U-Mart スタンドアローン版　市場分析ツール

このツールによって市場全体の動きが簡単に分析できるようになったが、個々のエージェントの動きを捉えるまでには至らなかった。そこで次に私が開発したのが 26 頁に示したツールである（図 5)。60 日間にわたる 1 体のエージェントの動きを追うことができる。売りと買い（Sell/Buy）どちらの注文をしたのか、注文価格（LimitPrice）はいくらか、注文数量 (Volume) はどのくらいか、そして注文したうちどのくらい約定して（ContractedVolume)、またどのくらい約定できずに次の板寄せへと持ち込まれたのか（RemainingVolume）という情報を示してくれるようにした。これによって特定のエージェントが行っている細かい動きと、市場全体の大きな動きの比較が可能になったが、すべてのエージェントの動きを 1 度に分析するツールはできていない。エージェント同士の相互作用がどのように働いているかということを分析するまでのところには踏み込めていない。

以下に示したのが member1 の取引状況を result.log から抽出して CSV 形式で保存する

ことができる U-Mart 解析ツールでのコードである。作成は私が行った。このツールは、分析をする上では何ら問題はないのだが、エクセル上の一部に「null」が出てしまうバグが発生することがわかった。そのため今回の実験では、このツールを使用しないことにした。

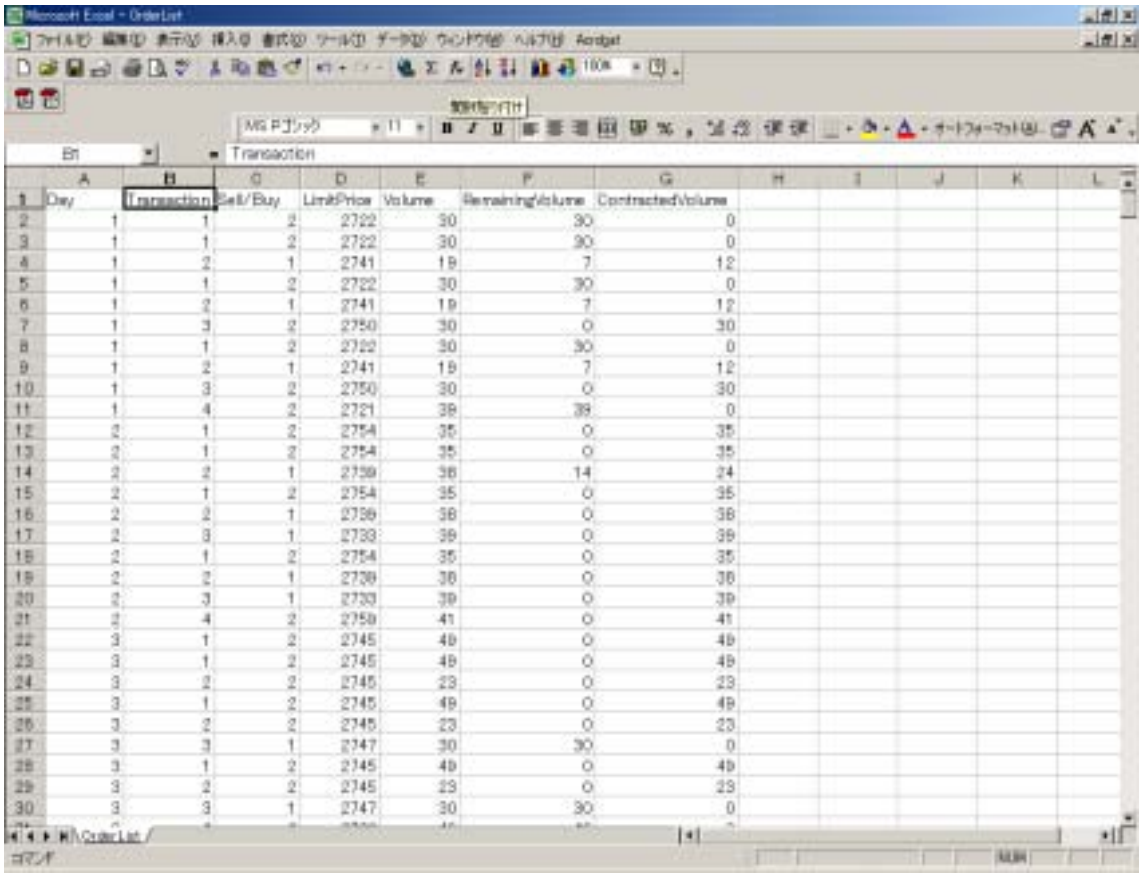
```
import java.util.*;
import java.io.*;
/**
 *        U-Mart 解析ツール
 *    member1 の取引状況を result.log から抽出して CSV 形式で保存する
 *
 *        @author Shigeto Kobayashi
 */
public class OrderReader {

   private static final int DAY = 60;
   public static String[] order = new String[DAY * 4 * 90];
   public static String orderLine;
   public static int count = 0;
   public static int endIndex = 0;

   public static void main (String[] args) {
      //result.log から取引状況を読み込む
      try {
      BufferedReader readOrder = new BufferedReader(new FileReader("result.log"));
         while ((orderLine = readOrder.readLine()) != null) {
            StringTokenizer tk = new StringTokenizer(orderLine," ");
            endIndex = tk.countTokens();
            for (int i = 0; i < endIndex; i++) {
               try {
               if (orderLine.startsWith("OrderID") == true) {
                  order[count] = tk.nextToken();
                  count++;
                  }

                  } catch(NoSuchElementException e0) {
                  System.out.println("No Token");
```

```
                }
            }
        }
        readOrder.close();
    } catch(FileNotFoundException e1) {
        System.out.println("No file");
    } catch(IOException e2){
    System.out.println("Can not Open");
  }
    try {
        FileWriter writeOrder = new FileWriter("OrderList.csv");
        //以下の 7 つの情報だけを切り取ってファイルに書き込む
        writeOrder.write("Day,Transaction,Sell/Buy,LimitPrice,Volume,
        RemainingVolume,ContractedVolume¥n");
        for (int i = 0; i < (DAY * 4 * 3) ; i++) {
            writeOrder.write(order[5 + (22 * i)] + ",");
            writeOrder.write(order[7 + (22 * i)] + ",");
            writeOrder.write(order[11 + (22 * i)] + ",");
            writeOrder.write(order[13 + (22 * i)] + ",");
            writeOrder.write(order[15 + (22 * i)] +"," );
            writeOrder.write(order[17 + (22 * i)] + ",");
            writeOrder.write(order[19 + (22 * i)] + "¥n");
        }
        writeOrder.close();
    } catch(FileNotFoundException e1){
    System.out.println("No file");
    } catch(IOException e2){
    System.out.println("Can not Open");
    }
  }
}
```



	A	B	C	D	E	F	G
1	Day	Transaction	Sell/Buy	LimitPrice	Volume	RemainingVolume	ContractedVolume
2	1	1	2	2722	30	30	0
3	1	1	2	2722	30	30	0
4	1	2	1	2741	19	7	12
5	1	1	2	2722	30	30	0
6	1	2	1	2741	19	7	12
7	1	3	2	2750	30	0	30
8	1	1	2	2722	30	30	0
9	1	2	1	2741	19	7	12
10	1	3	2	2750	30	0	30
11	1	4	2	2721	39	39	0
12	2	1	2	2754	35	0	35
13	2	1	2	2754	35	0	35
14	2	2	1	2739	38	14	24
15	2	1	2	2754	35	0	35
16	2	2	1	2739	38	0	38
17	2	3	1	2733	39	0	39
18	2	1	2	2754	35	0	35
19	2	2	1	2739	38	0	38
20	2	3	1	2733	39	0	39
21	2	4	2	2759	41	0	41
22	3	1	2	2745	49	0	49
23	3	1	2	2745	49	0	49
24	3	2	2	2745	23	0	23
25	3	1	2	2745	49	0	49
26	3	2	2	2745	23	0	23
27	3	3	1	2747	30	30	0
28	3	1	2	2745	49	0	49
29	3	2	2	2745	23	0	23
30	3	3	1	2747	30	30	0

図 5　U-Mart スタンドアローン版　エージェント分析ツール

U-Mart がはきだしてくれる大量の情報をいかにして分析していくのかという道具が簡単であるが完成した。私はここが到達点であるとは思っていない。個々のエージェントの動きをもっと詳細に分析できるツールの作成が今後の目標だと考えている。その前提としても、現段階で自分たちができることが何であるのかということを明確にするため、完成した自前の実験分析ツールを使って次のような実験を試みることにした。

第5章 実験デザインと結果の概要

U-Mart では基本的に 60 日の取引を 1 回としている。この日数は簡単に変更することができるのであるが､今回も分析を容易にするために 60 日を 1 回の取引をして扱うことにした。そしてこの 60 日の取引を 100 回行うことによって分析を行う。実験の公平性を保つために使用する現物の価格系列はランダムに選択した。しかし、エージェントを入れ替えて再度実験を行うことが予測されるので、使用する現物の価格系列の番号はすべて記録しておいた。価格変動として用いる「価格差」には｜p(t)-p(t -1)｜を採ることにした。実験には 10 の取引戦略をもつエージェントをそれぞれ用意。前章で紹介したサンプルエージェント(「裁定取引」「現物ランダム」「先物ランダム」「トレンド」「アンチ・トレンド」「現物移動

平均」「先物移動平均」「デイ・トレード」「相対力指数（RSI）」)に加えて、私が作成した「サイコロジカル・ライン」を 参加させた。

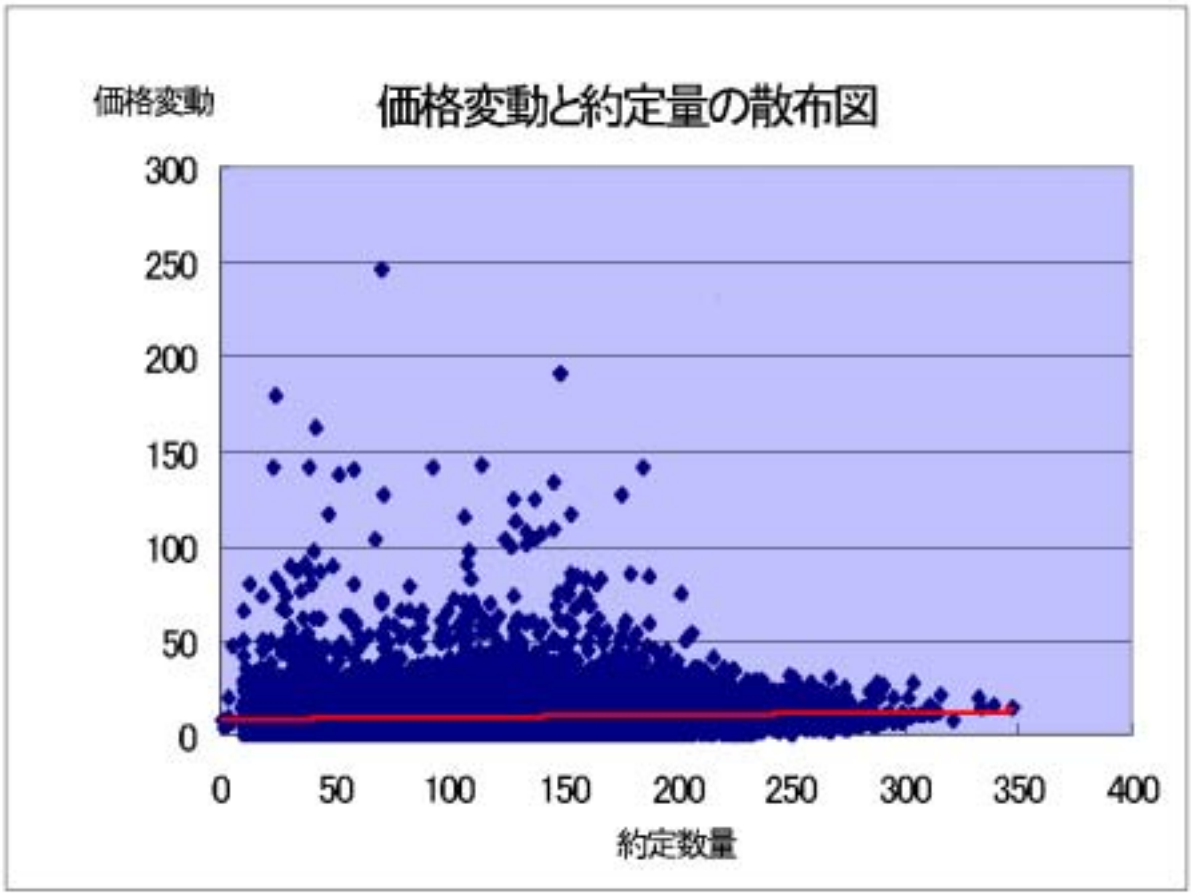
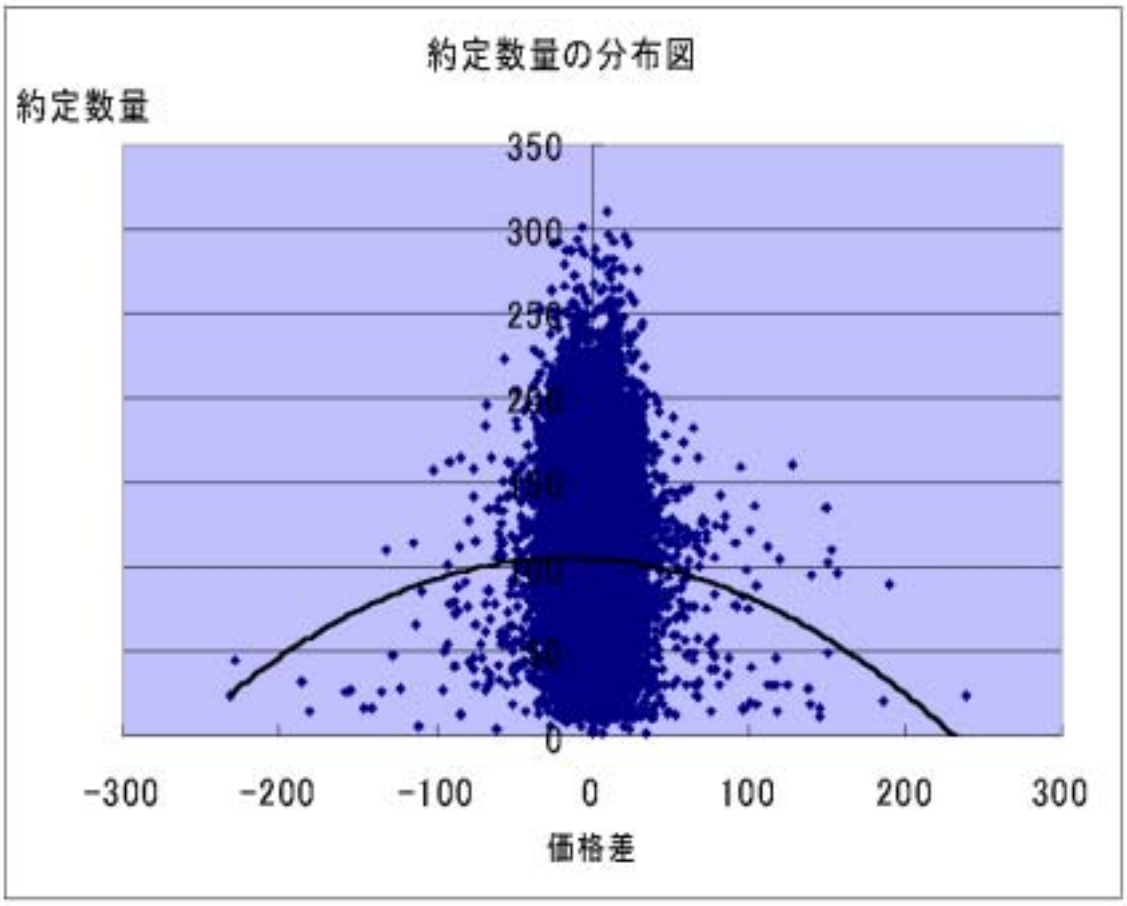


図 6　価格変動と約定量の散布図

＜実験１＞

縦軸に価格変動｜p(t)-p（t-1）｜をとり、横軸に約定数量をとった散布図を作成した。100 回分の実験を行ったため、対象となるサンプルが 23000 以上となり、データが密集してこのグラフでは見にくいものとなった。肝心のグラフの方は、かなり弱いながらも価格変動と約定数量には正の相関があることが確認された。ちなみに検定は有意水準５％で採択されている。このグラフからもわかるように 50 を超えるような大きな価格変動はそれほど起こっていない。ほとんどのサンプルが価格変動 50 以内で約定していることがわかる。これは採用する現物の価格系列にも影響されるのであろうが（意図的に暴落などを起こしてはいない)、実験参加エージェントの中に市場を荒らすような取引戦略を持っているものがいなかったことが安定的な市場になった理由と考えられる。また 200 を超える約定数量の方へ目を向けると、この部分も価格変動が 50 以内である。価格変動の小さい方が約定数量が大きくなるのはどうしてであろうか。私は価格変動が小さいときは需給が比較的安定していて約定されやすい、しかし価格変動が大きいほど、需給が一方に傾いて約定数量が減ってしまうと推察した。次頁の図 7 を見てもらおう。

図 7　約定数量の分布図



明らかに価格差が大きいときには約定数量が極めて少ない。反対に約定数量が多いのは価格差が小さい部分に密集している。これは私の推察がある程度正しいといえる範囲のものだと考える。だが、必ずしも価格差が大きいときだけ約定数量が少ないということではない。価格差が小さいときにも約定数量が少ないことが多いのである。全体的なグラフの形状としてはサンプル数の多さもあるが、価格差 0 を頂点とする棒状もしくは山形といえるであろう。だがこうした市場はエージェントの数やエージェントの組み合わせに依存する部分があるのではないか。エージェントの数はこの U-Mart のスタンドアローン版では変更することができないが、参加するエージェントは入れ替えることができる。ここまでの経緯を受けて東京工業大学の小山友介先生から次のような指摘を受けた。「ランダムエージェントがいる限り最低の注文量が常に確保されるのではないか」と。確かにそう述べることができるであろう。前章で記述したものの繰り返しとなるが、ランダムエージェントの基本的な取引戦略は次の通りである。

ランダムエージェントの取引戦略

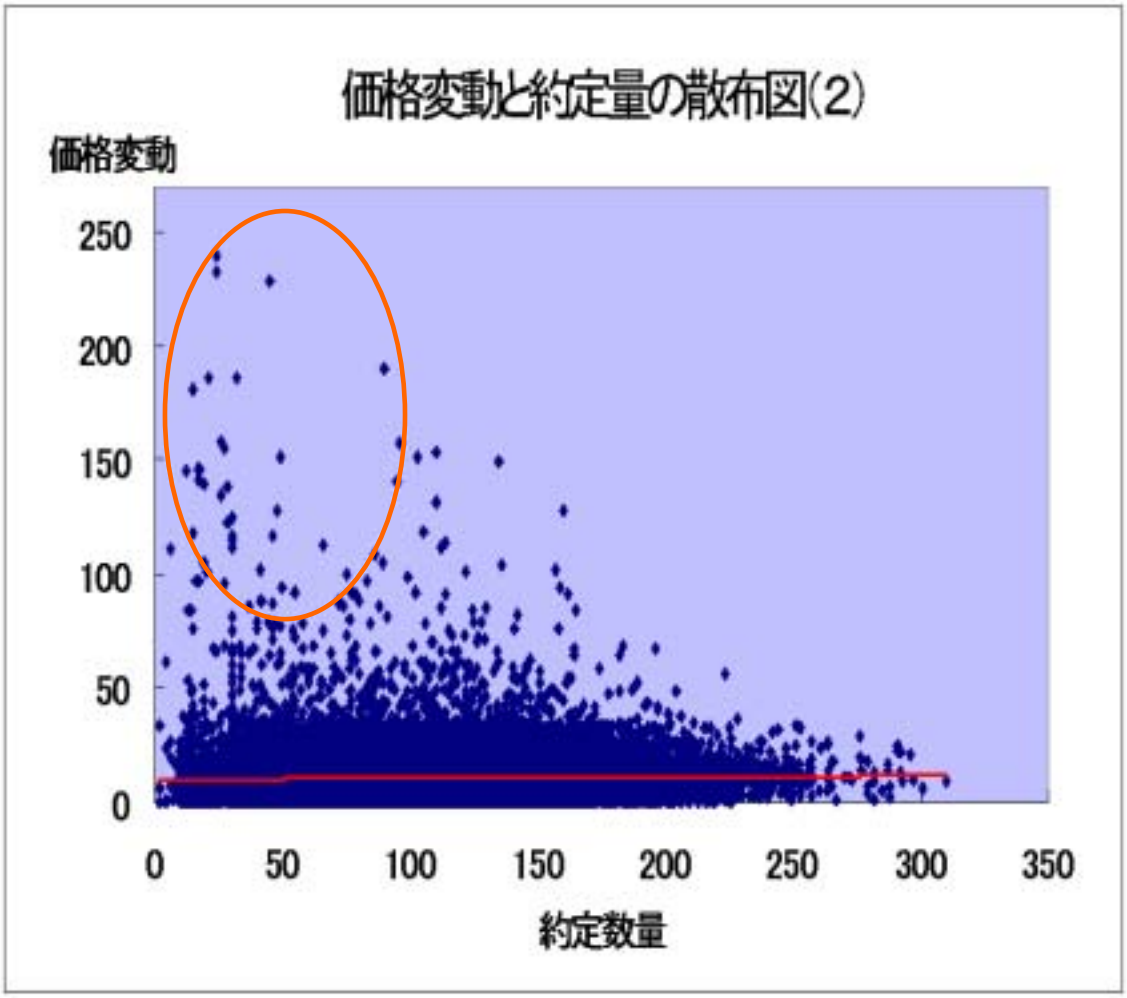
０が出たら⇒　取引しない１が出たら　⇒　売り２が出たら　⇒　買い

ランダムエージェントは現物を指標とするもの、先物を指標とするものの２つがある。さらにそれぞれの取引戦略エージェント（サイコロジカル・ラインは除く）は２体ずつ参加しているので計４体のランダムエージェントが存在することとなる。ランダムエージェントは、価格変動はおろか、何の影響を受けずに注文を出す。注文を出さない場合もあるが、4 体がそろって注文を出さない場合の確率は小さい。よって価格変動に関係なく最低限の注文が市場に出されていることがわかる。ではこうしたエージェントが取り除かれた場合の市場の動きはどのように変化するのであろうか。

＜実験２＞

ランダムエージェントを除いてストキャスティックエージェント(詳細は3章を参照)を入れる。ストキャスティックエージェントを入れたのは取引条件が価格に依存しており、ランダムエージェント以外の他のサンプルエージェントと比較しても獲得する利益水準に差がないことからである。実験の条件は前回とまったく同じにし、現物の価格系列も同じものを採用して実験を行った。

図 8　価格変動と約定数量の散布図（実験２）



前回の実験と比較して散布図（図 8）の形状には大きな変化はなく、価格変動と約定数量にも非常に弱いが正の相関があった。これも検定を通っている。実験１と比較して変化があったのは、大きな価格変動が多くなったことである。注目していた散布図左上のサンプル数が（橙色の楕円）増加した。実験条件は実験１とまったく変えていないので、ランダムエージェントを除外してストキャスティックエージェントを入れたことが影響していると考えられる。その原因はランダムエージェントがいなくなったことにより、価格変動に左右されずに常に最低限保たれていた安定的な需給がなくなったからではなかろうか。それによって価格が売買判定に影響するエージェントばかりになった市場は、前回の実験と比べて多少荒れ気味になったと考えられる。別の視点から両実験を見ることにしよう。

下図（図 9）は両実験での約定した期間を比較している。ランダムエージェントを抜いた実験 2 では、実験１よりも不約定の回数が 300 回あまり多かった。1 期における平均約定数量は 100 単位であるので、100 回分の実験での 300 回の不約定は合計で 30000 単位ほど差が出てくることになる。実験 2 で約定率が大幅に落ち込んだことからランダムエージェントは市場の流動性に寄与していると考える。実験２で価格変動が大きくなったのもこのことと関連があると考えている。

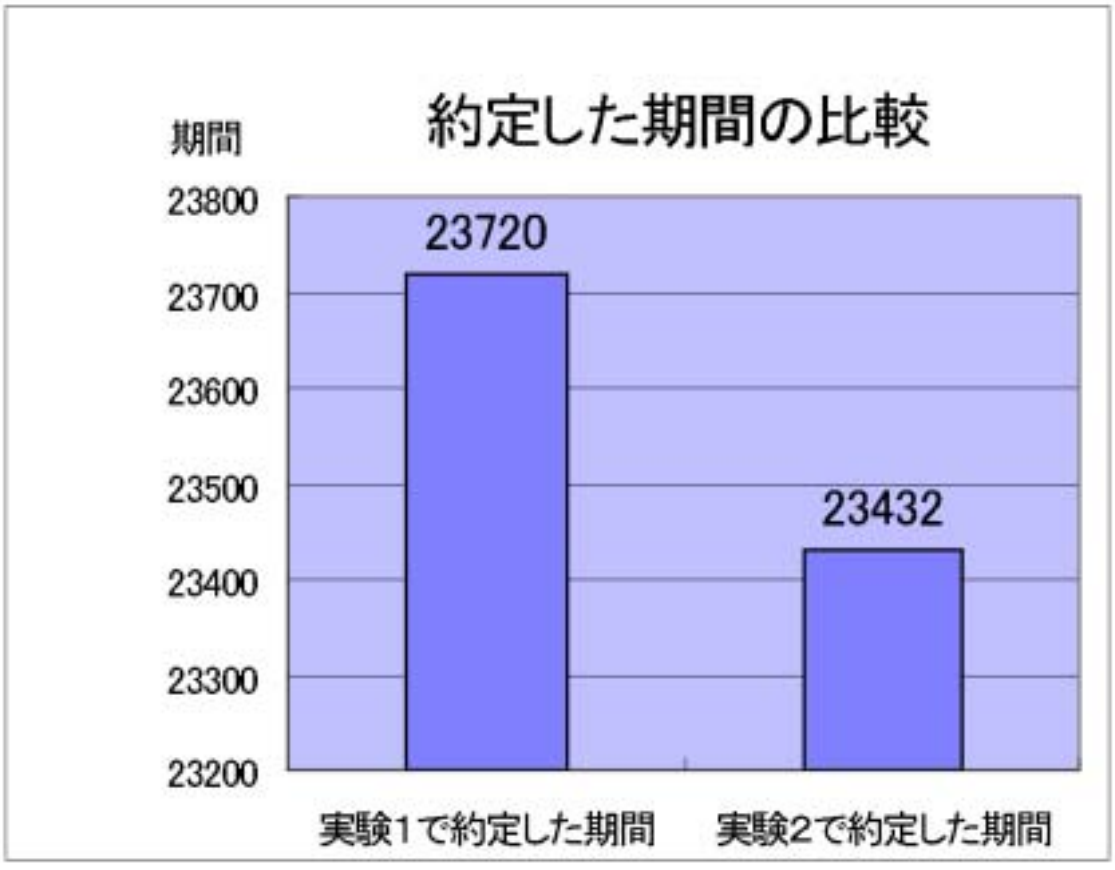


図 9　両実験での約定した期間の比較

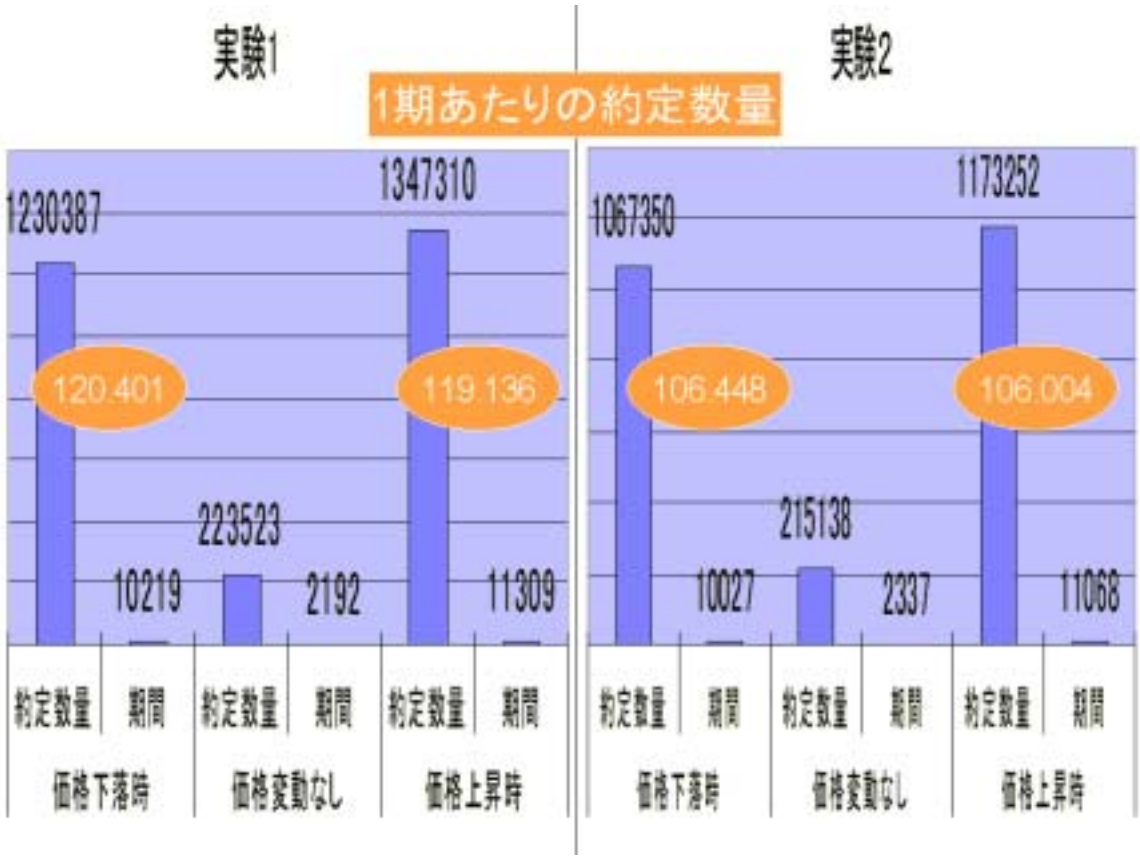


図 10　両実験の 1 期あたりの約定数量

さらに両実験を約定数量から眺めることにした。特に注目したのは価格上昇時・下落時における約定数量である。現実の株式市場では価格上昇時の方が約定する量が多いとされている。U-Mart ではどうであろうか。単純に数量だけを比較すると価格上昇時の方が価格下落時よりも約定数量が多い。しかし、採用する現物の価格系列に価格上昇傾向が強いと、総量では必ず価格上昇時の方が約定する量が多くなるはずである。そこで、それぞれの約定数量の総数とその期間を割ることによって 1 期あたりの約定数量を出し、価格上昇時と価格下落時で比較を行った。それが上図のオレンジ色で囲まれた数字である。明らかに両者の間には大きな差異はない。若干価格下落時の方が価格上昇時に比べて 1 期あたりの約定数量が多くなっているが、これはほとんど気にする程の数字ではないであろう。現実の市場と比較してこのような結果になったわけであるが、こうした結果になったのには私が敷いたいくつかの要因があると考えられる。

まず順張りと逆張りで売買方式を変えるエージェントが存在しない。また価格の上下にともなう心理的なウエイトを売買判断に組み込めていない。こうしたエージェントを組み込んで、今回の結果が変化するのかどうかはまだ実践できてはいない。既存のサンプルエージェントと先ほど挙げたようなエージェントの比率を組み換えて、何回か実験すること

により、市場おけるマクロでの動きがどのあたりで大きく変化するのかという部分に私は非常に興味を持った。これは今後の課題である。価格上昇時と価格下落時での 1 期あたりの約定数量にはほとんど差がないこと、このことも今回新たに発見したことである。これはランダムエージェントがいる、いないに関わらず同じ結果が得られた。両者の売買は鏡に映しだされたものであり、価格が上がっているときの売買は、価格が下がっているときの売買の単に逆の事象が行われているだけといえるのであろうか。この問題もこの先検証を行っていく予定である。

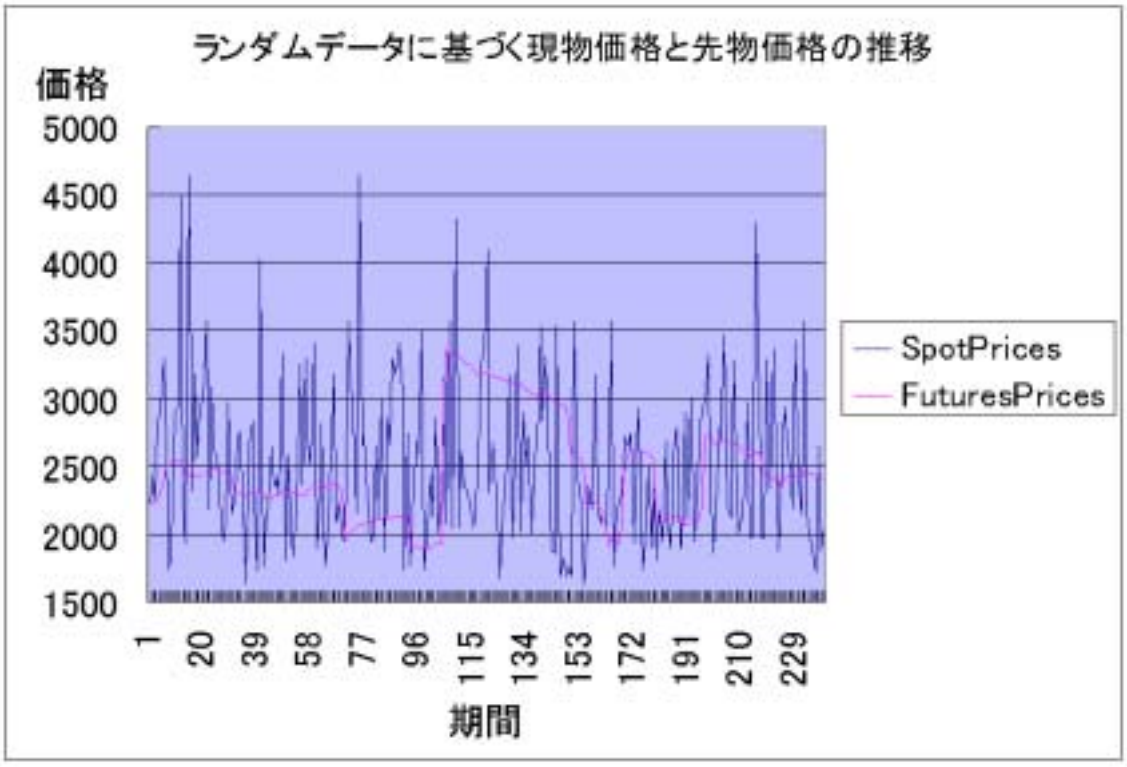


図 11　現物価格にランダムなデータを採用した場合の先物価格の推移

＜実験３＞

上の図（図 11）は私が現物価格を操作して意図的に急騰や暴落を起こしたときの先物価格の推移を表したものである。市場に参加させたエージェントは実験２と同じものを使用している。ある程度の乱高下であれば、U-Mart の先物価格も現物価格と同じ動きをすることが実証実験からわかっている。しかし、今回私が行った実験では乱高下があまりにも極端に激しすぎると先物価格が現物価格を追いきることができないという結果になった。もちろんこれだけの乱高下が起きているので、不約定になる回数も他のものと比べて多かったが、思っていたほどは多くは発生しなかった。現段階でこのような状態に何故なるのかは詳しくは掴んではいない。さらに個々のエージェントの動きを詳細に分析する必要性があるだろう。

おわりに

この実験は、2 人の人物がいなかったら、私の力だけではおそらく実行不可能であったであろう。1 人は実験解析ツールの大部分を開発してくれたゼミの後輩である石山洸くんである。彼とは Java のさわりも知らなかったころから、2 人 3 脚でエージェント開発を行ってきた。今となっては彼の開発能力は私のそれを遥かに超えてしまったが、Java に不慣れな私をやさしくサポートしてくれた。もう 1 人は東京工業大学の助手である小山友介先生である。まったく何も知らない私のために、いつもわかりやすく指導してくださり、実験が行き詰まったときには助言もいただいた。この 2 人には感謝の気持ちでいっぱいである。この場を借りて御礼申し上げたい。そして、いっかいの学部学生である私にいつも多くの人たちとの出会いを提供して下さった指導教授である有賀先生にも 2 年間の感謝の気持ちを示したい。

この論文を執筆するに至っては、私の研究不足の感が否めなかった。多くの部分で問題を先送りにし、この卒業論文できちんとした形で結論を導くまでにはならなかった。しかし、卒業論文を作り上げるという過程の中で、たくさんのことに気がつかされたのもまた事実である。私はこの研究は非常に可能性を秘めていると信じている。私の中で未だに多くのことが消化されてはいないが、この研究の新しい流れに加わっていけるようにこれからも勉強を続けていきたいと思っている。

参考文献

1. 人工知能って何，人工知能学会 HP　http://www.ai-gakkai.or.jp/jsai/
2. 和泉潔，植田一博，人工市場入門，人工知能学会誌，Vol.15，No6，pp.941～950，2000.
3. 和泉潔，個人の複雑さ、市場の複雑さ，情報処理学会研究報告 2001-ICS-123，pp19～24，2001.
4. 和泉潔，植田一博，人工市場と実験市場の統合を目指して，第 26 回システム工学部会研究会報告，pp17～23，2002.
5. 有山貴信，中島智晴，岩淵久生，学習型 U-Mart エージェントの設計と意思決定支援への応用，第 26 回システム工学部会研究会発表報告，pp13～16，2002.
6. エコノ探偵団，経済学って何に役立つ，日本経済新聞 2002 年 11 月 3 日朝刊紙面
7. W.Brian.Arthur, John.H.Holland, Blake LeBaron, Richard Palmer and Paul Tayler ‘Asset Pricing Under Endogenous Expectation in an Artificial Stock Market’, pp.1～4，1996.
8. 大村敬一，宇野淳，川北英隆，俊野雅司，株式市場のマイクロストラクチャー，日本経済新聞社，275ｐ，1998.

9. J.O'Neil 著，トップスタジオ訳，武藤健志監修，独習 Java，翔泳社，529p，1999.
10. 高橋麻奈，やさしい Java，ソフトバンクパブリッシング，551p，2000.
11. 河合昭男，オブジェクト指向がわかる，技術評論社，p214，2001.
12. ロイターリミテッド著，小島秀雄訳，小川真路訳，テクニカル分析入門「ロイター・ファイナンシャル・トレーニングシリーズ」日本語版，経済法令研究会，290p，2001.
13. 塩沢由典，U-Mart 計画：到達点と今後の課題，U-Mart2002 講演，2002.
14. 中島義裕, U-Mart を用いたこれからの研究計画, 進化経済学会大阪大会報告, 2002.
15. 寺野隆雄，U-Mart プロジェクト：人工市場研究から制度設計へ，進化経済学会大阪大会報告，2002.
16. 佐藤浩，小山友介，U-Mart2001 実験報告および既存エージェントの分類，進化経済学会論集，第６集，pp315～322，2002.

引用文献

1. 小林重人，石山洸，The Summary on Our U-mart Agent Programs，2002.
2. 小林重人，U-Mart2002 マシンエージェント解説，2002.
3. 中田翔，U-MART2002 エージェント解説，2002.

プログラミングコード

1. 小林重人，石山洸，A）the Arbitrage Transaction Strategy　B) the combination of Arbitrage Transaction Strategy and Williams %R Strategy　C) a Variant of the B D): the combination of Arbitrage transaction Strategy, Williams'%R Strategy and Spot Moving Strategy，May 2002.
2. 小林重人，Fluctal Psychological Line (Psychological Strategy)，October 2002.
3. 中田翔，原田圭，石山洸，Stochastic Strategy1・2，October 2002.
4. 石山洸(作者)，小林重人(改訂)，PriceReader 現物価格と先物価格を result.log から抽出して CSV 形式で保存する U-Mart 解析ツール（小林の改訂により約定数量も保存可能），November 2002.
5. 小林重人，OrderReader member1 の取引状況を result.log から抽出して CSV 形式で保存する U-Mart 解析ツール，November 2002

U-Mart サーバ

U-Mart Project　URL http://www.u-mart.econ.kyoto-u.ac.jp/