JAIST Repository
https://dspace.jaist.ac.jp/

Title	クラシファイアーシステムを用いた非同期型ネットワークゲームの作成
Author(s)	藪，記行
Citation	
Issue Date	2000-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/669
Rights	
Description	Supervisor:櫻井　彰人，知識科学研究科，修士

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

修　士　論　文

クラシファイアーシステムを用いた
非同期型ネットワークゲームの作成

指導教官　　櫻井彰人　教授

北陸先端科学技術大学院大学
知識科学研究科知識システム基礎学専攻

850096　藪 記行

審査委員：　櫻井　彰人　教授（主査）
林　　幸雄　助教授
橋本　　敬　　助教授

2000 年 2 月

Copyright _ 2000 by Noriyuki Yabu

目　次

1.　はじめに　**1**
1.1　研究背景　1
1.2　目的　2

2.　ゲームに関する考察　**4**
2.1　ゲームの面白さ　4
2.1.1　遊びの 4 分類　4
2.1.2　面白さの定義　7
2.1.3　有効性　11
2.2　ネットワークゲーム　12
2.2.1　ユーザフレンドリーなシステム　12
2.2.2　固定ゲームと発展ゲーム　12
2.2.3　同期システムと非同期システム　13
2.3　まとめ　13

3. 実装　**14**
3.1　設計　14
3.1.1　ゲーム概要　16
3.1.2　システム説明　18
3.1.3　クラシファイアーシステム　19

3.2　コーディング　22
3.2.1　戦闘処理プログラム　22
3.2.2　アカウント操作プログラム　29
3.2.3　部品表示プログラム　36
3.2.4　ランキング表示プログラム　37
3.2.5　ウェブページ　38
3.3　実験　39
3.3.1　実験 1　39
3.3.2　結果 1　39
3.3.3　考察 1　40
3.3.4　実験 2　41
3.3.5　結果 2　42
3.3.6　考察 2　42
3.3.7　実験 3　43
3.3.8　結果 3　44
3.3.9　考察 3　45
3.4　まとめ　46

4.　おわりに　**47**
4.1　評価　47
4.2　今後の展望　48

謝辞　49

参考文献　50

第　1　章
は じ め に

1.1　研究の背景

近年、数百万人もの人々が家庭でゲームを楽しんでいる。ゲームソフトはプレイステーションで週におよそ 10 本前後も出荷され、ユーザはその中から気に入ったソフトを購入して遊ぶ。これらゲームソフトの多くは ROM という形態で一方的にユーザに供給されては消費されていく。ここでいうゲームの消費とは一定の期間が経過してしまったり、一回エンディングまで到達してしまうと、プレイヤがやる気を無くしてしまって繰り返しプレイされることがない状況を指す。ゲームが消費される理由として、大きく二点あると考えられる。一つ目はゲームそれ自体が面白くないこと。二つ目は ROM という供給形態により、不具合に対応できないことや飽きられてしまうことが挙げられる。

ゲームが面白くない原因としては、ゲームの難易度が適切ではなかったり、単調であったり、ストーリーに興味を持てないなど、いろいろ考えられる。そこでゲームの面白さがどのような要素からなっているのか分析し、その要素をゲームに取り込むことで結果としてゲームは面白くなると推定できる。

ゲームプレイが飽きられてしまうのは、何度もゲームをプレイしても同じ内容であったり、不具合や理不尽さを修正できない ROM という形態が原因であると考えられる。ROM という形態ではなく、例えばインターネットなどによるアップデートが可能となれば、不具合やプレイヤの要望に対応できる。その結果、飽きられることを回避できると考えられる。

面白く、かつ飽きないゲームは消費されないゲームであると言える。

1.2　目的

本研究の目的は、

①ゲームを面白くする要素を分析し、その要素を組込んだゲームを作成する。そしてその結果ゲームが面白くなったか調べる

②ネットワークゲームという供給形態を採用する

である。

ゲームのどのような要素が面白さにつながっているのか調べ、ゲームの面白さを判断する 8 つの指標を提案する。その 8 つの指標を満足するには、どういった機能をゲームに実装すればいいのかを述べる。またネットワークゲームという供給形態を採用することから、そのゲームシステムはどうあるべきか考察する。8 要素の中から実現可能な 5 つの指標をゲームに実装することと、システム面から考えた設計で、1 対 1 でロボット同士を戦わせるゲームを作成する。その主な機能的な特徴は以下のとおりである。

①インターネット対応

②サーバ･クライアント方式

③非同期システム

④クラシファイアーシステムを実装

⑤ランキング制を導入

⑥インターネットブラウザのみでゲームが楽しめる

⑦電子掲示板での情報交換可能

インターネットで対戦することにより、現実の距離に関係なく、多くの人と一緒にゲームをプレイできる。インターネット利用者も年々増えているので、十分なユーザの確保ができると考えられる。またインターネットゲームは公開するのに、ソフトを出荷したりする必要はなく、1 人～少数でも開発、公開が容易にできる利点がある。ゲームバランスが悪い場合でも、プレイヤのフィードバックを得やすく、かつゲームに反映し易い。

ゲームシステムはサーバ･クライアント方式を採用し、データの整合性を保つ。また、ゲームシステムを改善する場合にも、直接サーバ側のプログラムを変更するので、ユーザに必ず反映される。

インターネットの回線速度が遅い人でも楽しめるように、非同期システムを導入する。非同期システムとは、現実世界の時間で同時に接続せずともゲームが進行できるシステムで、プレイヤ同士のネットワークへの接続タイミングの差にゲームの進行が影響を受けない方式である。

プレイヤの予想を覆す反応をゲームシステムに実装したい場合、簡単な方法として乱数を使う手法があるが、それだけではゲームは面白くならない。そこで、戦闘プログラム内に GA の手法の 1 つであるクラシファイアーシステムを用い、ロボットに行動パターンを学習させる。戦闘で学習しながら戦うので、意外な結果を引き起こすことも考えられる。動的に行動パターンを変化させるゲームは他になく、新規性を満たすと考える。

戦闘の結果はランキング制により全員が見ることができるので、さらにプレイするやる気を出させる。1 プレイが比較的短時間で終了するゲームは気軽に始められ、繰り返しプレイできるという点から、反復性を持つと言える。反復性をもったゲームは中毒性があり、長く遊んでもらうことができる。

パソコン初心者には未だにインストールという作業は難解と考え、ゲーム自体は専用なプログラムなどを必要とせず、最低限インターネットブラウザでゲームが楽しめるように、静的な部分は HTML で、動的な部分は CGI で実装する。

電子掲示板を設け、情報交換など、他のプレイヤとのコミュニケーションが可能である。リアルタイムではないシステムの場合は、この方式が一般的でかつ効果的であると考える。また、書き込む早さや生活時間帯の差に関係なく、多くの人が利用できる。

これら、競争の概念、コミュニケーション、ゲームバランス、意外性、新規性などがゲームを面白くすると考える。

第　2　章
ゲ　ー　ム　に　関　す　る　考　察

2.1　ゲームの面白さ

この章では、ゲームの面白さとはいったいどういう要素からできているのか、また客観的に面白いと言えるゲームとはどういったものなのか、またゲームシステムとはどうあるべきなのかを述べる。なお、ここで述べているゲームとはコンピュータゲーム全般を指すものであり、PC ゲームや家庭用ゲームであることを特に問わない。ゲームの面白さを比較するには客観的な指標が必要になる。そして、その項目とはどういうものにしたらよいのかが問題となる。本研究ではロジェ･カイヨワの「遊びと人間」を参考にした。

2.1.1　遊びの 4 分類

カイヨワは遊びの基本的な定義を述べ、遊びを 4 つに分類した[1]。

外面的　･競争(アゴン)
　　･運(アレア)
内面的　･模擬(ミミクリ)
　　･眩暈(イリンクス)

外面的な要素として競争(アゴン)と運(アレア)の 2 分類、内面的な要素として模擬(ミミクリ)と眩暈(イリンクス)の 2 分類をあわせて 4 分類である。

アゴンとはすべて競争という形をとる一群の遊びである。チェスやサッカーなどがこれに当たる。遊びの原動力は、どの競争者にとっても、一定の分野で自分の優秀性を人に認められたいという欲望である。そのため、アゴンの実戦は不断の注意、適切な訓練、たゆまぬ努力、そして勝利への意志を前提とする。アゴンは個人的能力の純粋形態として現れ、それを表明するのに役立つ。

アレアとはラテン語でサイコロの遊びを意味する。アゴンとは正反対に、遊戯者の力の及ばぬ独立の決定の上に成り立つすべての遊びを示す。ここでは、相手に勝つよりも運命に勝つことの方がはるかに問題なのだ。言い換えれば、運命こそ勝利を作り出す唯一の存在であり、相手のある場合には、勝者は敗者より運に恵まれていたというだけのことだ。この遊びの典型的な例はサイコロ、ルーレット、裏か表か、宝くじなどである。運そのものがゲームの勝敗を左右する。

アゴンとアレアは相反する、いわば対称的な態度を表わしているが、しかしどちらも同一の掟に従っている。それは、現実にはありえない純粋に平等な条件を、遊戯者間に人為的に作り出すという掟である。というのは、実生活においては何一つの明確なものはなく、ただ明らかなのは、幸運も功績も、すべてその始めは混沌としているというだけなのだ。そのため、アゴンであれ、アレアであれ、遊びは現実生活の状態である混乱に代えて完璧な状況を作り出そうとする試みである。

ミミクリは架空の環境において活動を展開したり、運命に服従したりするところにではなく、遊ぶ人間が架空の人物となり、それにふさわしく行動するというところに成立しうる。人が自分を自分以外の何かであると信じたり、自分に信じこませたり、あるいは他人に信じさせたりして遊ぶ、という事実にこれは基づいている。劇の上演と演技が当然このグループにはいることは明かである。

イリンクスに含まれるのは、眩暈の追求に基づくもろもろの遊びである。それらは、一時的に知覚の安定を破壊し、明晰であるはずの意識をいわば官能的なパニック状態におとしいれようとするものである。すべての場合において、一種の痙攣、失神状態、あるいは茫然自失に達することが問題なのである。声を限りに叫ぶこと、傾斜を走りおりること、回転滑り台、ジェットコースター、十分高くあがった場合のブランコ、これらは類似の感覚を与える。

これらすべての遊びを 2 つの相反する極の間に配置することもできる。一方の極においては、気晴らし、騒ぎ、即興、無邪気な発散と言った共通の原理が、ほとんど例外なしに支配している。そこには統制されていない気まぐれと言ったものが感じられるが、カイヨワはこの原理を、パイディアと呼んでいる。反対の極においては、この奔放ないたずらな気分は、消滅し、あるいは少なくとも、馴致され、代わって別の傾向が現れる。それは、この無秩序で移り気な性質とは相補関係にある傾向、すべてにわたって、というわけではないが、いくつかの点でこれと正反対の傾向のことである。恣意的だが強制的でことさら窮屈な規約に遊びを従わせ、いっそう面倒な障害を設けて遊びを縛る必要があるのは、無秩序で移り気な性質を安易に目標に到達させないためである。目標といってもまったく無用なものに変わりないが、それでいていっそうの努力、忍耐、技、器用がなければこの目標に到達はできない。カイヨワはこの第二の極をルドゥスと名付けた。

これら遊びの分類を表に表わすと、表 1 のようになる。

	アゴン (競争)	アレア (運)	ミミクリ (模擬)	イリンクス (眩暈)
パイディア (遊戯) 騒ぎ はしゃぎ ばか笑い 凧あげ 穴送りゲーム トランプ占い クロスワード ルドゥス (競技)	規則なし ・競争 ・取っ組み合い 運動競技 ボクシング フェンシング チェス サッカー スポーツ競技 全般	鬼決めじゃんけん 裏か表か遊び 賭け ルーレット 宝くじ	子供の物真似 空想の遊び 人形、おもちゃの武具 仮面 仮装服 演劇 見世物全般	子供の ぐるぐるまい メリーゴーランド ブランコ ワルツ スキー 登山 空中サーカス

注) 縦の各欄内の遊びの配列は、上から下へパイディアの要素が減少し、ルドゥスの要素が増加していくおおよその順序に従っている。

表 1　遊びの分類

2.1.2　面白さの定義

人はなぜ、ゲームに熱中のするのだろうか。その答として、「つまりテレビゲームとは、コンピュータによって人間たちの遊びが凝縮され、かつ編集されたものである」と平林久和は述べた[3]。そして、有名なドラゴンクエストを取り上げ、カイヨワの遊びの 4 分類のすべての要素を満たしているか検証した。冒険者となって剣を振るうところや、架空の街、海、森、洞窟など、人の会話や買い物などの行為が「模擬」、体力が 0 になると死んでしまう戦いは「競争」、その戦いの中でしばしば起きる"会心の一撃"は「運」で、見知らぬ土地を探索する場面などは「眩暈」と説明している。本研究ではこれら模擬、競争、運、眩暈といった指標はゲームを面白くする上で重要であると考え、着目した。

そこで一般的に、これら 4 つの要素はゲームにどのように適用されているのか、4 要素をゲームの指標として考察する。またそれら要素がゲームを面白くする上で重要であることを述べる。

まず「模擬」だが、ゲームをするということは、すでに仮想の世界がその中にあり、その目的も現実世界とは違っている。コントローラ等を操作することでゲームの中の世界に働きかけている点で一般的にゲームというものはすべてこの要素を満たしていると考えられる。しかし、たとえ仮想世界でもその世界観が理不尽であったりしてはいけない。また、新しいゲームシステムであっても世界観が過去のゲームの流用であったり、二番煎じであると、面白さは損なわれる。面白さを評価するサブ項目として、キャラクター性、ストーリー性、新規性を挙げる。

ゲーム業界を例に取ると、キャラクター性やストーリー性を満たすには、ゲーム制作者ではなくキャラクターを描くことが専門のイラストレーターや物語を書くことが専門の作家を起用したりする。

新規性はあればいいというわけでもなく、うまくゲームにマッチした内容であることが面白さにつながる。最近では一見ゲームと関係ないものをゲームシステムに適用することで新規性をだす場合もある。例えばプレイステーションのモンスターファームである。このゲームは音楽 CD やデータ CD-ROM などから、モンスターを生み出すという特徴を持っている。この手法はバーコードからあるデータを作り出して戦わせる、バーコードバトラーなどと同じ性質の面白さを生み出すと考えられる。

「運」は推理ゲーム等のロジカルに進めていくゲームでは使われていないことが多いが、他のゲームには欠かすことのできないくらい重要な要素となっていることが多い。この要素が無ければ、プレイヤのある入力に対して同じ反応しか示されないため、多くの人がプレイしても同じ経路･結末であるので、再度プレイしてもらう機会を失う。この要素のサブ項目として、ゲームバランス、意外性を挙げる。

ゲームバランスとは、

①ロールプレイングゲームならイベントごとの一区切りにかかるプレイ時間やその難易度

②シューティングゲームなら敵が打つ弾の数や、敵の壊れづらさ

③対戦ゲームならコンピュータの操作するキャラクタの強さ

これらがゲームを楽しむ上で適切であるかどうかの評価基準の総称である。ゲームバランスはゲームをプレイする上で適度な基準値でなけらばならない。難易度が対象の場合は簡単だと張り合いがなくつまらないし、難しいとやる気が失せてしまう。また、プレイに要する時間が長すぎると疲れたり、ゲーム途中で飽きてしまう。この要素はゲームを楽しむ上でもっとも重要な要素の一つであり、ゲームバランスが悪いと俗に言う「クソゲー」の汚名をこうむる。

もう一つのサブ項目である意外性はゲームで実現するのは難しい。これはゲームシステムがあるアルゴリズムによって記述されているため、ゲームをある程度やりこんだ人なら、その行動や反応はある程度予測可能になる。この要素をゲームに実装する簡単な手法は相手に人間を使うことである。その他の手法としてはニューラルネットなど人工知能を実装している例もある。例を挙げればプレイステーションの「がんばれ森川君 2 号」がある。

「競争」は遊びの本質的なものであり、ゲームにも適用されている。ゲームは大きく分けて 1 人用と多人数用に分けられるが、1 人でプレイする場合には競争の相手はゲーム内のコンピュータが代用しているものが多い。多人数でプレイする場合にはこの「競争」を主な目的としたもの、例えばレースゲームやテーブルゲームなどがある。他に共同作業でコンピュータと競争するものもある。また、カイヨワは遊びは 1 人で遊ぶ遊びも次第に社会性を持つと主張している[2]が、例えば 1 人用のロールプレイングゲームなどの場合は同じゲームを行う仲間が競争相手になっている場合がある。どのくらいゲームを進めたのか、レアなアイテムは持っているか、などである。

他にはプレイヤ同士でチームを組んで他のチームと戦ったりする。この現象がまさに遊びの社会性である。この要素のサブ項目として、コミュニケーション、競争の概念を挙げる。

オンラインゲームと呼ばれるコミュニケーションを重視したゲームがあるが、このようなゲームはチャット機能というものが実装されていて、リアルタイムでの対話が可能となる。また、電子掲示板でも情報交換が十分に可能であり、コミュニケーションがとれると考えられる。現に SEGA はドリームキャストでのインターネット接続を家庭用ゲーム機で初めて実現し、ゲームソフトごとに電子掲示板を用意してコミュニケーションを行えることを可能にしている。ここでは情報交換が活発であり、1 人用のゲームにおいてもディスカッションや雑談が行われている。カイヨワの主張している遊びの社会性はこのような電子掲示板でも実現できると考える。

競争の概念はゲームを楽しくするには必要不可欠で、対戦ゲームなどではこの要素を前面に押し出してる。SEGA のバーチャファイターに代表される対戦格闘ゲームと呼ばれるゲームが典型的な例であり、ときおり大会なども開かれる。競争の概念がなければその競争は単調になり、すぐに飽きられてしまう。逆にこの競争の概念が存在するゲームであれば、すぐに飽きることはなく、長くそのゲームを楽しんでもらうことができる。この要素を実現するには、プレイヤが選べる選択肢が多すぎず、少なすぎない適度な数を用意しなくてはならない。適当数ある選択肢の中でプレイヤがその都度考え選択することで駆け引きを生み、それが対戦を面白くする。じゃんけんは選択肢の数が絶妙な究極のシステムであると考える。この３すくみシステムは、ゲームの要素に取り入れられていることが多い。対戦格闘ゲームの 1 つであるデッド･オア･アライブを例に挙げれば、攻撃、投げ、いなしの 3 つがこのゲームの最大のポイントとなっている。攻撃は投げに強く、投げはいなしに強く、いなしは攻撃に強いということである。

「眩暈」について、カイヨワはスキーやジェットコースターを例に挙げている。ゲームでは高度な視覚効果を使用する場合が該当すると考える。ポリゴンを使用したリアルなグラフィックや、ゲームの途中に挿入されるムービーなどである。また映像だけではなく、音や振動など、臨場感をかき立てる要素ならばすべてが関与している。これら自体にはルールや意志は特になく、ただ見ているだけで楽しかったり、刺激になったりする。この要素のサブ項目として、エンタテイメント性を挙げる。

今までに無かったようなグラフィックを導入したり、今まで表現できなかったカメラワークであったり、ハードウェアが進化するに従って制限がなくなり、現在ではさまざまな表現が可能になってきている。他には 3D 音響設備や、家庭用ゲーム機のコントローラの振動システム、ゲームセンターにある画面の動きに合わせて椅子が動いたりする筐体でこの要素は実現される。これらの刺激が今までに無いものである場合、それだけでゲームを面白くする。

これらをまとめると以下のようになる。

・「模擬」

　　キャラクター性

　　ストーリー性

　　新規性

・「運」

　　ゲームバランス

　　意外性

・「競争」

　　競争の概念

　　コミュニケーション

・「眩暈」

　　エンタテイメント性

これら 8 項目によってゲームは面白くなると考える。

逆に、ゲームを評価する場合にこの 8 項目について調べることにより、ゲームが面白いか判断できると推測できる。

2.1.3　有効性

前章で述べた 8 項目を使い、いくつかのゲームを評価してみる。今回取り上げたのは、ロールプレイングゲームの代表作であるファイナルファンタジーⅠとⅦ、格闘ゲームとして一世を風靡したバーチャファイター 1 と 3、ネットワークゲームでは UltimaOnline とライフストーム、「落ちゲー」といわれるまでに有名になったテトリスとぷよぷよである。

主観ではあるが、この 8 種類のゲームを実際に評価したものを以下の表に示す。これを見てわかるとおり、ゲームの種類によってどういう部分が面白いのかがよくわかると思う。また、同じゲームでも他のゲームとどういった点で差別化をはかっているのかがよくわかる。

	FF1	FF7	VF1	VF3	UO	LS	テトリス	ぷよぷよ
キャラクター性	○	○	○	○	○	○	×	○
ストーリー性	○	○	○	○	○	○	×	○
新規性	◎	×	◎	×	○	×	○	×
ゲームバランス	○	○	○	○	○	×	○	○
意外性	×	×	○	○	○	○	×	×
競争の概念	×	×	○	○	○	○	○	○
コミュニケーション	×	×	×	×	◎	○	×	×
エンタテイメント性	×	◎	×	◎	○	×	×	×

表 2　ゲームの評価

雑誌等でゲームの面白さを評価するのはレビュアと呼ばれる人たちが複数人でそれぞれゲームの面白さを評価する手法をとっている。これと同じく、複数人で上に挙げた 8 つの項目において評価すれば、客観的な判断ができる。また全体で一つの点数をつけることはしない。プレイヤによって、そのゲームが面白いと思う要素は違うはずである。ロールプレイングゲームの場合は新規性などよりもストーリーを重視する人、格闘ゲームの場合そのゲームバランスが気になる人、パズルゲームにもキャラクタ性が欲しい人、高度なグラフィックやムービーを使っていてエンタテイメント性を感じればいいという人、などさまざまである。この 8 項目の指標に基づいて、ある一定数以上の人による評価を参考にすれば、客観性を示すことも可能であり、この 8 項目がゲームが面白いかどうかを判断する指標として有効であると

考える。

2.2　ネットワークゲーム

ゲームを構築するにあたって、面白いだけではゲームは成立しない。ユーザフレンドリーなインタフェースや多くの人が楽しめるシステム、長く遊んでもらう配慮などが必要である。

2.2.1　ユーザフレンドリーなシステム

パソコンのゲームの場合、家庭用ゲーム機にない、ソフトのインストールという作業がある。インストールという作業はパソコンの初心者にとって難解であり、できれば避けたい作業である。また、複雑な説明を必要とせず、直感的に操作できるものが好ましい。作業や操作を単純化することはそのゲームシステムの敷居を低くすることになり、より多くの人が楽しめると考える。

2.2.2　固定ゲームと発展ゲーム

長く遊んでもらうためには、ゲームの不具合に逐次対応したり、プレイヤの要望などに応える必要がある。家庭用のゲーム機の多くは ROM の状態で市場に出荷されるため、ゲームの不具合やプレイヤの要望は反映されない。バランスの悪さや要望などは葉書でのアンケートなどによってフィードバックされる。ROM のような書き換え不可能な媒体で提供されるゲームを固定ゲームと呼ぶ。固定ゲームの場合、ファイナルファンタジーⅠ～Ⅷのようなシリーズものでも毎回 ROM を出荷する必要があり、またユーザはその都度お金を払ってそのシステムを買わなければならない。そして古くなったゲームはプレイされなくなる。しかし、ゲームシステムがサーバなどで管理できる場合、ゲームシステムを改良することで不具合や要望を反映でき、システムそのものが新しく置き換わるので、プレイヤにとってはそのゲームをプレイする上で妨げとなっていた問題が取り払われることは嬉しい。ゲームを提供する側もシステムを更新するコストは安い。このように即時に更新可能なゲームシステムを発展ゲームと呼ぶ。オンラインゲームと呼ばれているゲームは典型的な発展ゲームであり、不具合や要望に対応している。例を挙げれば、Ultima Online が有名で、発売されてから 2 年半を数え、今ではプレイヤ人口は全世界で 50 万人に達している。Ultima Online は 2 週間～2 ヶ月くらいの周期でシステムが改良され、プレイヤの要望

も反映されている。このゲームではカウンセラーと呼ばれるボランティアのプレイヤが一般プレイヤからのフィードバックをゲーム開発者に伝える役割を担っている。ゲームを長期的に遊んでもらうためには発展ゲームである必要がある。

2.2.3　同期システムと非同期システム

ネットワークゲームには大きく分けて、同期システムと非同期システムがある。同期システムとは、ゲームをプレイする上で、現実の時間での同時接続により成り立つ。ピアツーピアという接続法式では必ず同期システムでなければならない。反対に、非同期システムはゲームをプレイする上で、相手と同時に接続していなくとも成立する。クライアントサーバ方式では、すべてサーバを介してシステムが管理されているので、プレイヤは他のプレイヤと同時に接続せずとも、サーバとの接続のみでゲームを楽しむことができる。あるプレイヤのプレイで、他のプレイヤのデータへの影響をサーバ側で処理できるシステムである。にそれぞれメリットとデメリットがあり、ゲーム内容によって使い分けられている。同期システムは Ultima Online のように、接続の時間帯を他のプレイヤと合わせる必要がある。レスポンスが早いことが大きなメリットだが、サーバとクライアントの間のデータ転送が遅い場合はゲームの面白さを損なう可能性がある。また現実世界の時間の制約を受けるというデメリットもある。非同期システムは同期システムとは相反する性質を持っており、回線の速さの違いはゲームシステムに関係しない、現実世界の時間の制約を受けにくいメリットがある。また、データはすべてサーバにあるので管理し易い。反面デメリットはレスポンスが遅いことだ。非同期システムとして有名なのは Lord of Monsters であり、web ゲームの中では根強い人気を保っている。現在、シリーズ四作目まで発展してきた。

2.3　まとめ

本研究ではこれら要素から実現可能な項目として、新規性、ゲームバランス、意外性、競争の概念、コミュニケーションを選択し、実際にゲームに実装することを目標とする。またインストールを必要とせず、最低限インターネットブラウザのみでゲームを楽しめるシステムにし、非同期システムで発展ゲームを実現する。

第　3　章
実 装

3.1　設計

この章では前章で述べた、ゲーム要素のうち実装可能な 5 項目をどのようにゲームに組み込むか解説する。

新規性、ゲームバランス、意外性、競争の概念、コミュニケーションという要素を満足するゲームシステムについて述べる。

新規性を満たすには今まで使われていないアイディアを実装する必要がある。本研究では「NPC 動作アルゴリズムの自動生成に関する考察」[4]を参考にした。この論文は、コンピュータゲームにおいて、敵の動作アルゴリズムを自動生成することについて述べている。ここでは対戦格闘型アクションゲームを例に挙げ、NPC(ノン･プレイヤ･キャラクタ)の動作を自動生成する手法として GA を用いたと説明している。

本研究で用いた、GA の手法の 1 つであるクラシファイアーシステムは実際にコンピュータゲームに実装されている例はない。本研究ではこのクラシファイアーシステムに着目し、これをゲームに実装することで新規性を満たすと考える。また、戦闘を行わせるたびに学習するので、意外性もあると考える。ゲームの主な遊び方はプレイヤ同士のロボットの対戦で、かつ勝率によるランキング制を導入することにより競争の概念を満たす。ゲームサイト内に電子掲示板を設け、ゲーム対する討論や情報交換でコミュニケーションを計る。さらに十分なテストプレイを繰り返してパラメータを調整することより、どのようなタイプのロボットでも学習次第では強くなるよう、適度なゲームバランスを実現する。またパラメータの偏りが激しい、というようなプレイヤからの報告からパラメータの調節も必要である。

これらゲームを面白くする要素と、ユーザフレンドリーなインタフェースや多くの人が楽しめるゲームシステム、長く遊んでもらう配慮などから本研究で作成するゲームの要求仕様を以下に示す。

・総合ゲームサイト(web ページ)
・クラシファイアーシステムを実装したロボット作成プログラムと
　戦闘処理プログラムによるテキストタイプの戦闘結果出力
・個人ランキング表示
・電子掲示板
・十分なバランス調整
・インターネットブラウザのみでゲームをプレイできる
・発展ゲーム
・非同期システム

3.1.1　ゲーム概要

本研究で作成するゲームはバーチャルなロボット同士を戦わせるのが主な内容である。ロボットの主な概要を図 1 に示す。

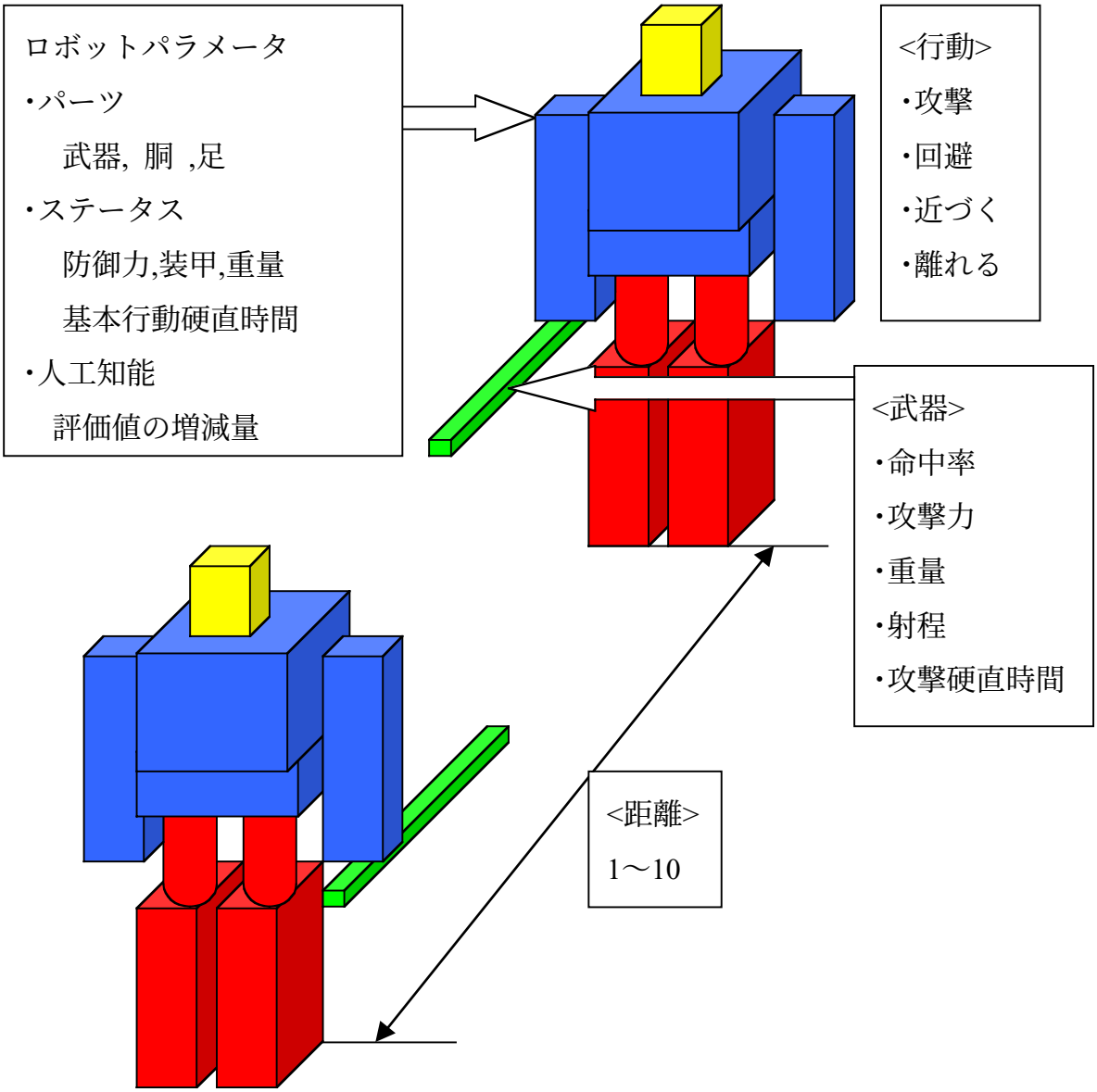

図 1　ロボットの概略図

プレイヤはロボットの名前を入力し、武器や胴体部品などの設定をカスタマイズする。そして他のプレイヤのロボットを指定して、戦闘を開始させると、人工知能が攻撃、回避、前進、後退といった行動を逐次選択し、相手の装甲を 0 にした方が勝ちとなる。ゲームの大まかな流れを以下に示す。

① アカウント登録

倉庫にてプレイヤの名前(オーナー名)、パスワード(アクセスコード)、ロボット名(電脳兵名)を入力し、武器、胴体部品、足部品を選択肢の中から選ぶ。さらに、ロボットは戦闘中学習するので、ロボットがとった行動の評価値をあらかじめプレイヤが与えることができる。これにより、ロボットの行動パターンを調整できる。これらを一通り入力し「登録する」ボタンを押すと、サーバにデータがセーブされる。

② 他のロボットと戦闘

闘技場にて、自分のプレイヤ名とパスワードを入力し、自分のロボットのステータスを確認する。その後対戦相手を選んで「戦闘開始」ボタンにより戦闘が開始される。直後に戦闘結果が HTML 形式で表示され、その戦闘における学習結果、勝敗数等が自分のデータに書き込まれる。

③ 調整

パーツやパラメータ設定が気に入らないところがあれば、倉庫にて変更する。

④ ②と③の繰り返し。

戦闘では、お互いのロボットが行動待ち時間という値を持ち、この値をそれぞれ同時に減らしていく。行動待ち時間の値が 0 になると行動が可能となり、攻撃、回避、前進、後退の 4 つのうちの１つ行動をとる。行動後はそれぞれの行動に設定された行動遅れ時間が次の行動待ち時間となる。攻撃を受けると装甲値が減り、0 になると負けとなっている。また一定時間に勝負がつかない場合は引き分けとなる。

ゲームにおけるパラメータを概説する。それぞれのロボットには攻撃力、防御力、武器使用時の行動遅れ時間、基本行動遅れ時間、武器命中率、武器品質、武器射程、重さ、装甲というパラメータをもっている。その他変数は経過時間、ロボット間距離が挙げられる。

戦闘のアルゴリズムに関しては後述する。

3.1.2　システム説明

すでに述べた要求仕様を満足するシステムのうち、直感的にわかりづらいと考える諸項目について概説していく。

十分なバランス調整を行うのは当然であるが、プレイヤからのフィードバックも有効に活用するシステムが要求される。本研究ではロボット対戦ゲームであるので、バランスを取るのが非常に難しい。そもそもゲームバランスがいいとは、適度な難易度であるとか最強の選択肢が存在しないかそれ自体がゲームの難易度に与える影響が少ない場合であると考える。最強の選択肢が存在してしまうと、対戦ゲームの場合はユーザの選択は最強パターンに多く集中してしまう。このような状況ではゲームの選択肢が狭まることになり、面白さが損なわれてしまう。理想はタイプの違うロボットすべてにおいて同じ難易度であるように何度も調整、テストを繰り返さなくてはならない。しかし選択肢が徐々に増えていくと、すべてのパターンで難易度を同程度にするのは時間コストがかかりすぎる。また、客観的に判断する指標というものが確立されていない。そこで、プレイヤーからのフィードバックを参考に調整するのである。現在ネットワークゲームと呼ばれるシステムは、正式リリース前にβテスタと呼ばれるボランティアのプレイヤを募り、漸次調整されていく。この方式はコストが安く、インターネットが普及した現在、最も有効であると考える。

発展ゲームであるには、上で述べたバランス調整を逐次行うことと、さらに、ゲーム内での新しい選択肢を増やすことが必要だと考える。それまでにない選択肢はプレイヤに新しい刺激を提供する。新しい刺激は、ゲームが常に新鮮であると感じさせるために必須であり、飽きさせないための仕掛けである。

3.1.3　クラシファイアーシステム

まず、本研究のシステムの原案となった NPC 動作アルゴリズムの自動生成に関する考察について簡単に概説する。この方法論では対戦格闘型アクションゲームへの実装を想定している。このタイプのゲームにおいては、NPC も PC(プレイヤ･キャラクタ)と同様の動作を行うので、人間のプレイヤがどのように状況を判断して操作を行うのかを参考に、動作アルゴリズムを設計できる。ゲーム内で操作によって行える動作は、数種類の攻撃、移動など限られている。そこで、可能な選択肢の中から、状況によって最適なものを選ぶわけであるが、ここでいう状況とは次のようなものから表現されると説明している。

･敵との水平距離
･フィールド場での位置
･敵の動作状態
･自キャラクタの動作状態
･残体力
･残時間
･その他

これら状況の各要素を以下、状況変数と呼ぶ。NPC 動作アルゴリズムは、書き下された手順として実装される場合もあるが、アルゴリズムをデータ列で表現し、様々な操作を行い易くする。

まず、アルゴリズムが状況(状況変数群)を入力、操作を出力とする関数の形式で表現できることを確認しておく。ここで、動作状態などの状況変数は、定義された数個の値しか取らない。例えば、動作状態では、移動中＝1、攻撃動作中＝2、ジャンプ中＝3 と定義されていれば、変数の値は 1、2、3 のいずれかである。

またコンピュータゲーム上で、整数で表現される距離などの状況変数も、実際には場合分け程度の表現で実用上問題がないと解説している。例えば、攻撃の当たる距離であれば、実際の距離が 1 でも 2 でも同じである。これら状況変数が有限個の離散値で表現可能と考えることは、操作が数個の離散値しかとらないことからも妥当である。

このモデルで遺伝アルゴリズム(GA)を用いて、強い動作アルゴリズムを得る手法を以下に示す。

動作アルゴリズムを表現した多次元配列を染色体とするモデルを考える。配列内の各変数が取り得る値は、移動、攻撃などのあらかじめ定義された有限個の値でしかないため、それをそのまま対立遺伝子とできる。次に、ランダムに構成された染色体の集合を用意する。染色体集合のメンバーに順列付けを行い、評価の低い染色体を破棄して、集合を更新する。新規に作成される染色体は、前の世代の生き残りの集団から、交叉や突然変異によって得られる。こうして染色体の順序付け、世代交代を繰り返す事で、優秀な染色体の獲得が可能であると解説している。

本研究ではプレイヤ同士のロボットを戦わせるゲームであるが、この戦闘でのロボットの動作アルゴリズムをクラシファイアーシステムを用いて表現する。

戦闘内における環境変数のうち、ロボットの行動そのものに影響を与える項目、敵の射程、敵の装甲、敵の動作、自分の射程、自分の体力の 5 項目である。敵の射程は敵の武器の射程とそのときのロボット間の距離から、近ければ 0、丁度良ければ 1、遠ければ 2 という整数値で表わす。以下 4 項目についても環境から整数値を割り当て、これらをつなげた 5 桁の数列を環境染色体と呼ぶ。戦闘のいかなる時点でもこの 5 桁の数列で記述可能である。環境染色体に記述されているある環境の時、とるべき行動を記述した 1 桁を加えた 6 桁の数列を行動染色体と呼ぶ(図 2)。ロボットは行動染色体を一定数もっており、戦闘中、自分が行動可能なある時点での環境染色体とのマッチングを取り、複数ある行動染色体を見つけ、最後の桁に記述してある行動を選択する。行動結果からその行動染色体の評価値を上下する。

環境染色体の全パターン数よりもロボットがもつ行動染色体の数の方が少なく、行動染色体に記述されていない環境が存在する。その時の行動はランダムで行い、行動染色体の中でも評価値が最も小さいものと入れ替わる。

戦闘中、ある一定時間ごとに新しい行動染色体の生成を行う(図 3)。行動染色体の中でも評価値が最も高い 2 つを取り出し、交叉を行う。交叉点は行動染色体の 5 カ所の接点のうち、その都度ランダムで決定する。交叉後できた新しい行動染色体は評価値が最も低い 2 つに置き換えられる。ここで交叉を行った場合にできた染色体がすでに行動染色体として記述されていた場合には、その染色体が唯一となるまで、突然変異を起こす。突然変異は 6 桁のうち、ランダムで 1 桁のみが変更される。

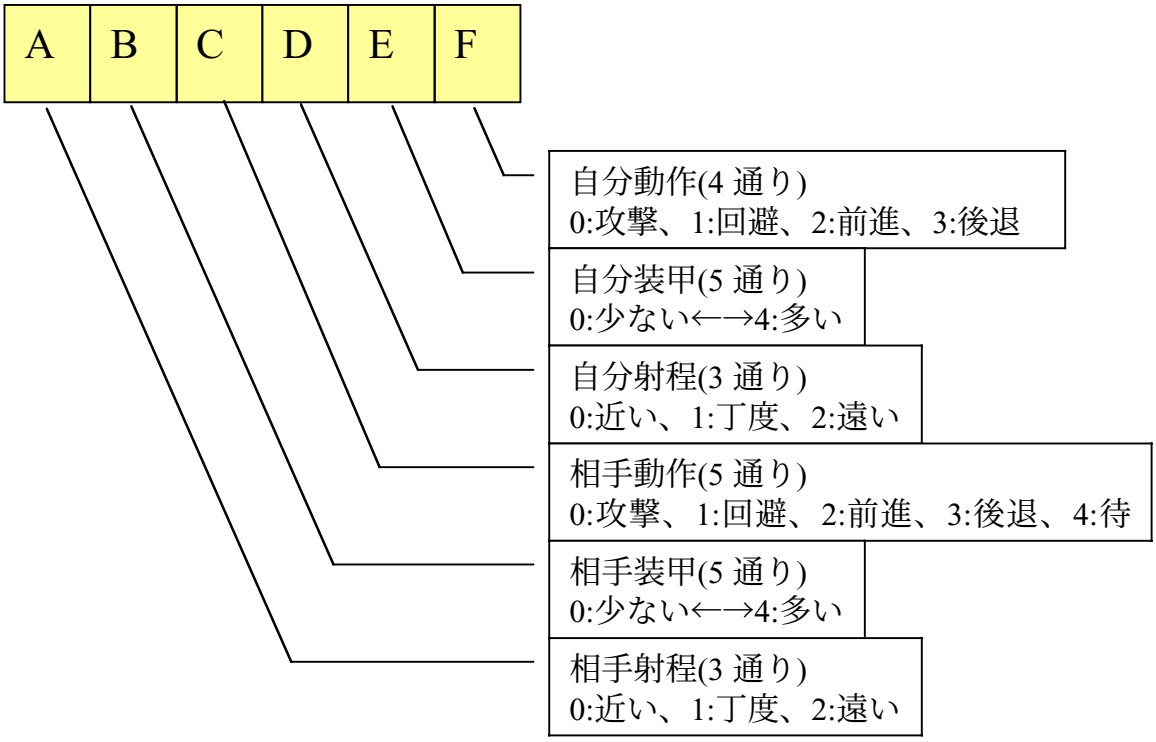

図 2　行動染色体の概念図

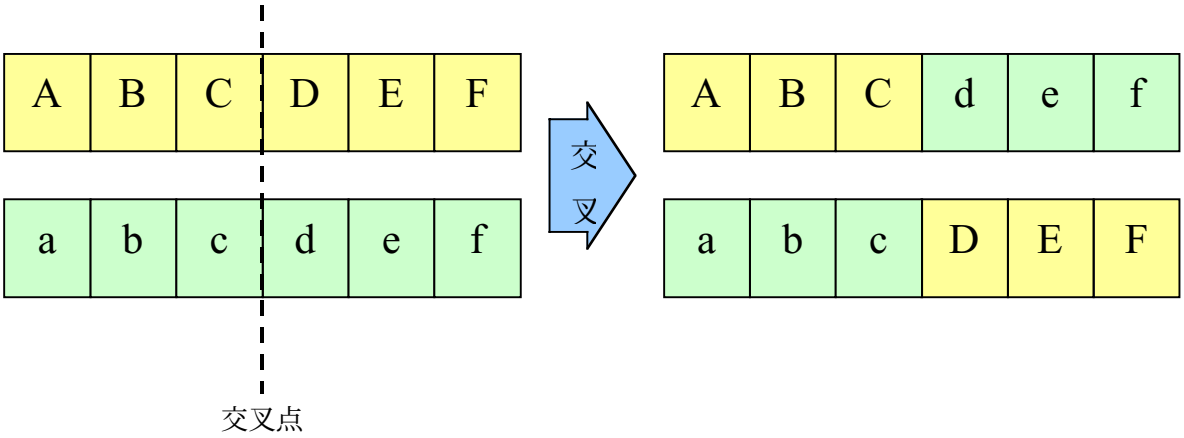

図 3　交叉の模式図

これら、①環境認識、②環境染色体生成、③行動染色体検索、④行動を繰り返し、時折交叉や突然変異を起こしながら、徐々に最適な行動染色体群を得る。

評価値の増減はある行動染色体から唯一決まるものではなく、その行動後の結果により変化する。このことから、戦闘を多く行い学習していくことによって強いアルゴリズムを獲得すると考える。

3.2　コーディング

ゲームシステムのうち､動的に変化させる必要がある部分は CGI として実装する。CGI は全部で戦闘処理プログラム、アカウント操作プログラム、部品表示プログラム、ランキング表示プログラムの 4 つである。

3.2.1　戦闘処理プログラム

ゲーム内では｢闘技場｣と呼んでいる。この CGI はアカウントデータファイルからプレイヤのロボット情報等を読み込み、対戦相手を選んで戦闘処理を行うプログラムである。戦闘結果は動的に変化し、HTML 形式で出力される。戦闘の内容は保存せず、クラシファイアーシステムで学習した結果や勝敗数などをアカウントデータファイルに書き戻し、終了する。以下に変数を示す。アカウントデータのフォーマットは 3.2.2 で述べる。

①データファイルから読み出した数値(左が自分で右が相手のプレイヤ)

オーナー名	$s_owner	$e_owner
電脳兵名	$s_robot	$e_robot
武器名	$s_weapon	$e_weapon
胴体部品名	$s_body	$e_body
足部品名	$s_leg	$e_leg
武器攻撃力	$s_power	$e_power
攻撃硬直時間	$s_delay	$e_delay
基本行動硬直時間	$s_slow	$e_slow
防御力	$s_armor	$e_armor
装甲	$s_life	$e_life
武器命中率	$s_hit	$e_hit
武器品質	$s_qual	$e_qual
全体重量	$s_weigh	$e_weigh
武器射程	$s_range	$e_range
行動染色体列	$s_chromo_file	$e_chromo_file

行動染色体	$s_gene	$e_gene
行動染色体の評価値	$s_gene_eval	$e_gene_eval
行動染色体配列	@s_chromo	@e_chromo

②戦闘中のみ使う環境変数

最後にとった行動	$s_action	$e_action
硬直時間	$s_wait	$e_wait
残り装甲	$s_life_rem	$e_life_rem
環境染色体	$s_chromo_env	$e_chromo_env
環境染色体配列	@s_env	@s_env
お互いの距離	$distance	
染色体配列の数	$gene_num	

③命中率の計算式

a=(武器射程-2)/武器射程

b=abs(相手との距離-武器射程)

c=武器命中率*b**a

if(相手の最後の行動＝攻撃){　c*=1.2　}

elsif(相手の最後の行動=回避){　c/=2　}

命中率=c+武器品質

以下に戦闘処理プログラムのフローチャートを図 4-a～4-e に示す。

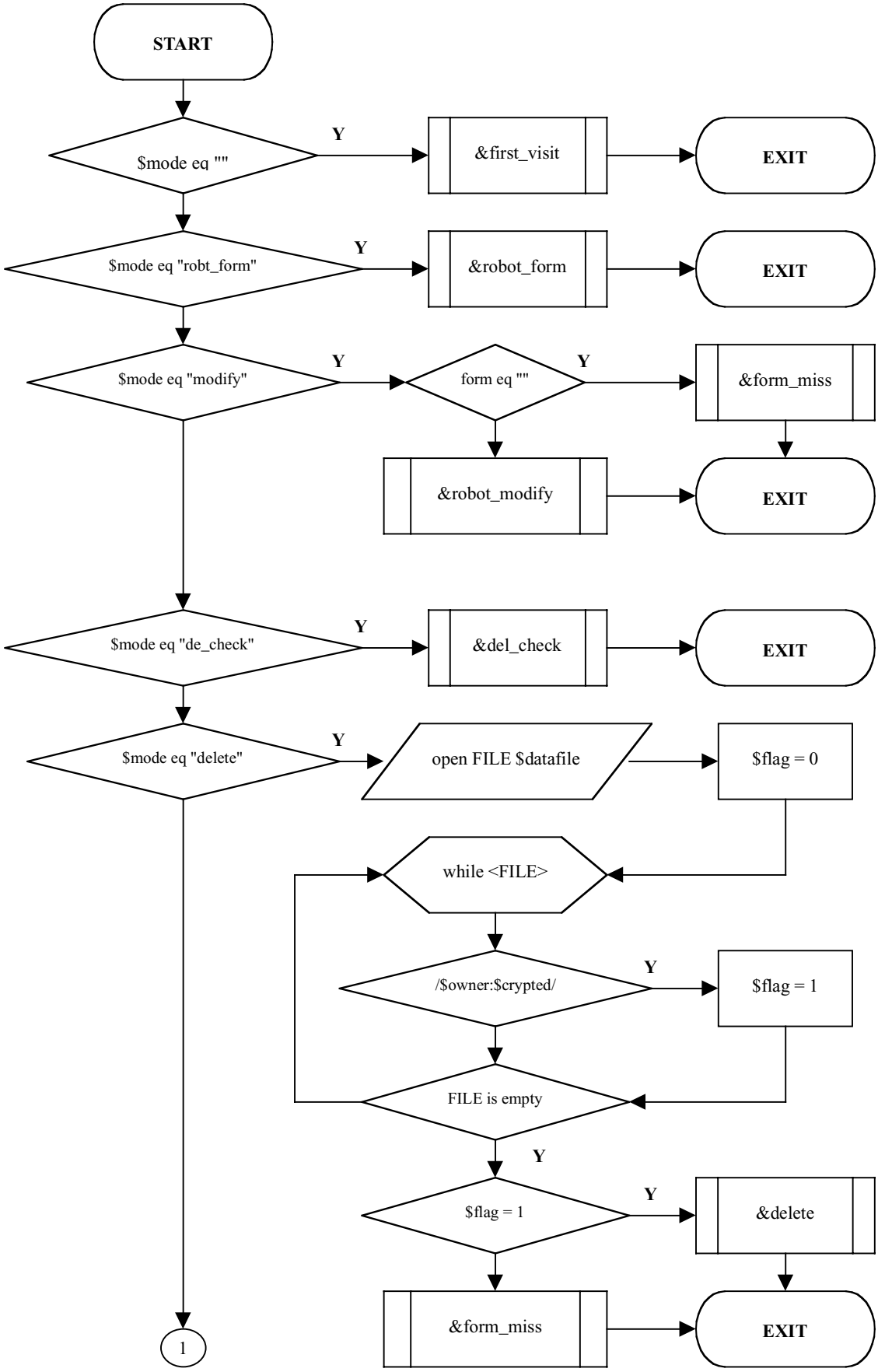
START
$mode eq ""
Y
&first_visit
EXIT
$mode eq "robt_form"
Y
&robot_form
EXIT
$mode eq "modify"
Y
form eq ""
Y
&form_miss
&robot_modify
EXIT
$mode eq "de_check"
Y
&del_check
EXIT
$mode eq "delete"
Y
open FILE $datafile
$flag = 0
while <FILE>
/$owner:$crypted/
Y
$flag = 1
FILE is empty
Y
$flag = 1
Y
&delete
&form_miss
EXIT
1

図 4-a

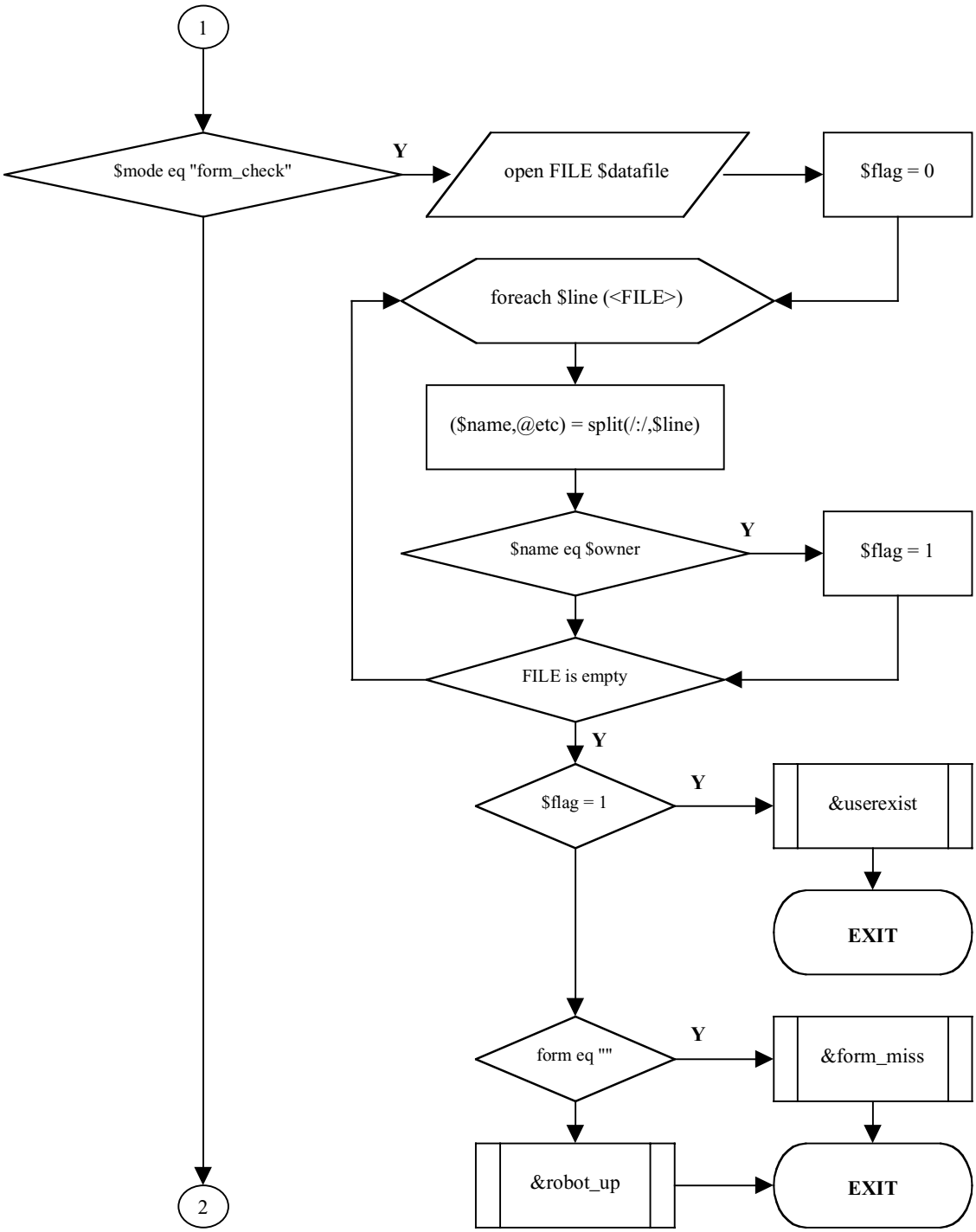
1
$mode eq "form_check"
Y
open FILE $datafile
$flag = 0
foreach $line (<FILE>)
($name,@etc) = split(/:/,$line)
$name eq $owner
Y
$flag = 1
FILE is empty
Y
$flag = 1
Y
&userexist
EXIT
form eq ""
Y
&form_miss
&robot_up
EXIT
2

図 4-b

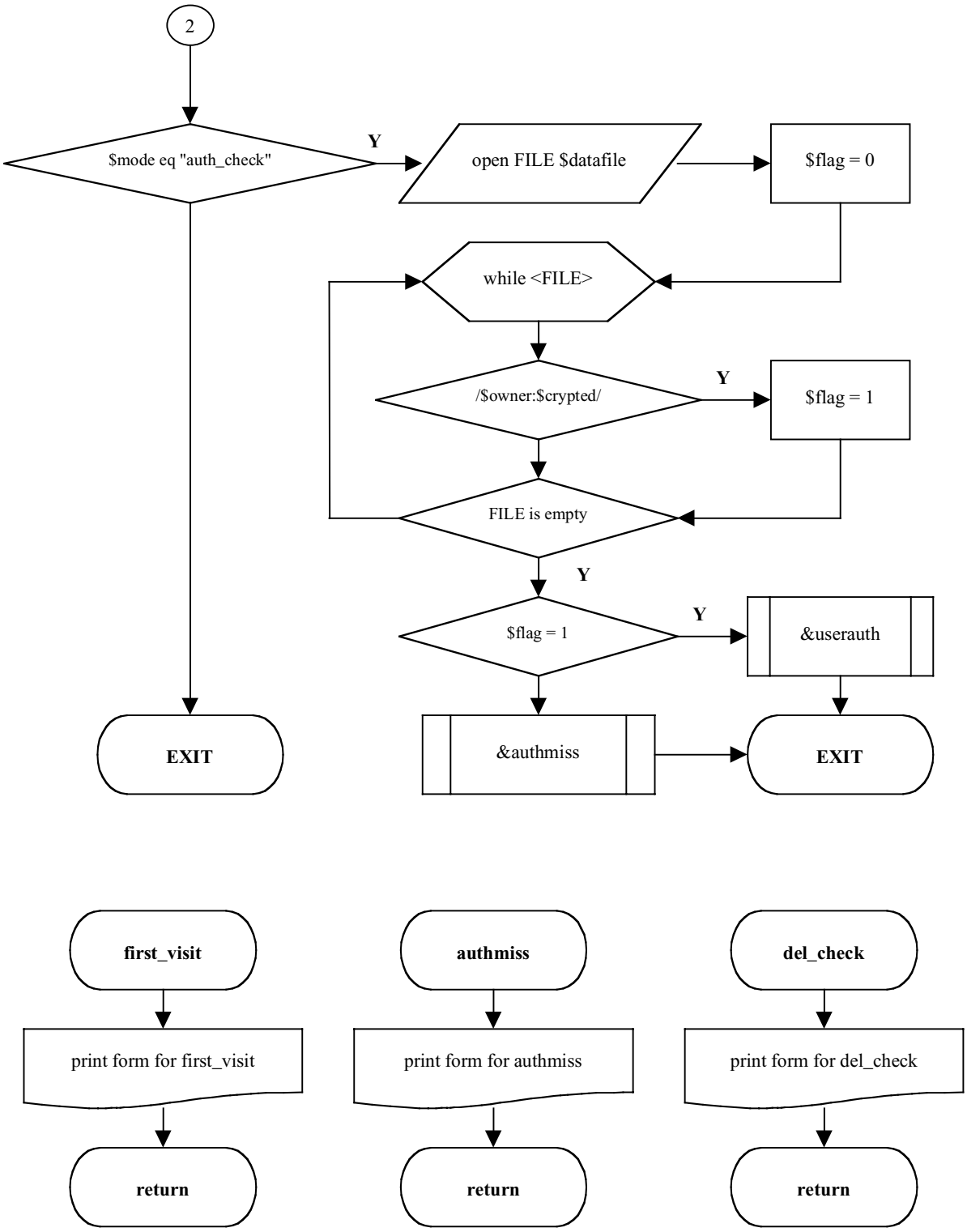
2
$mode eq "auth_check"
Y
open FILE $datafile
$flag = 0
while <FILE>
/$owner:$crypted/
Y
$flag = 1
FILE is empty
Y
$flag = 1
Y
&userauth
EXIT
&authmiss
EXIT
first_visit
print form for first_visit
return
authmiss
print form for authmiss
return
del_check
print form for del_check
return

図 4-c

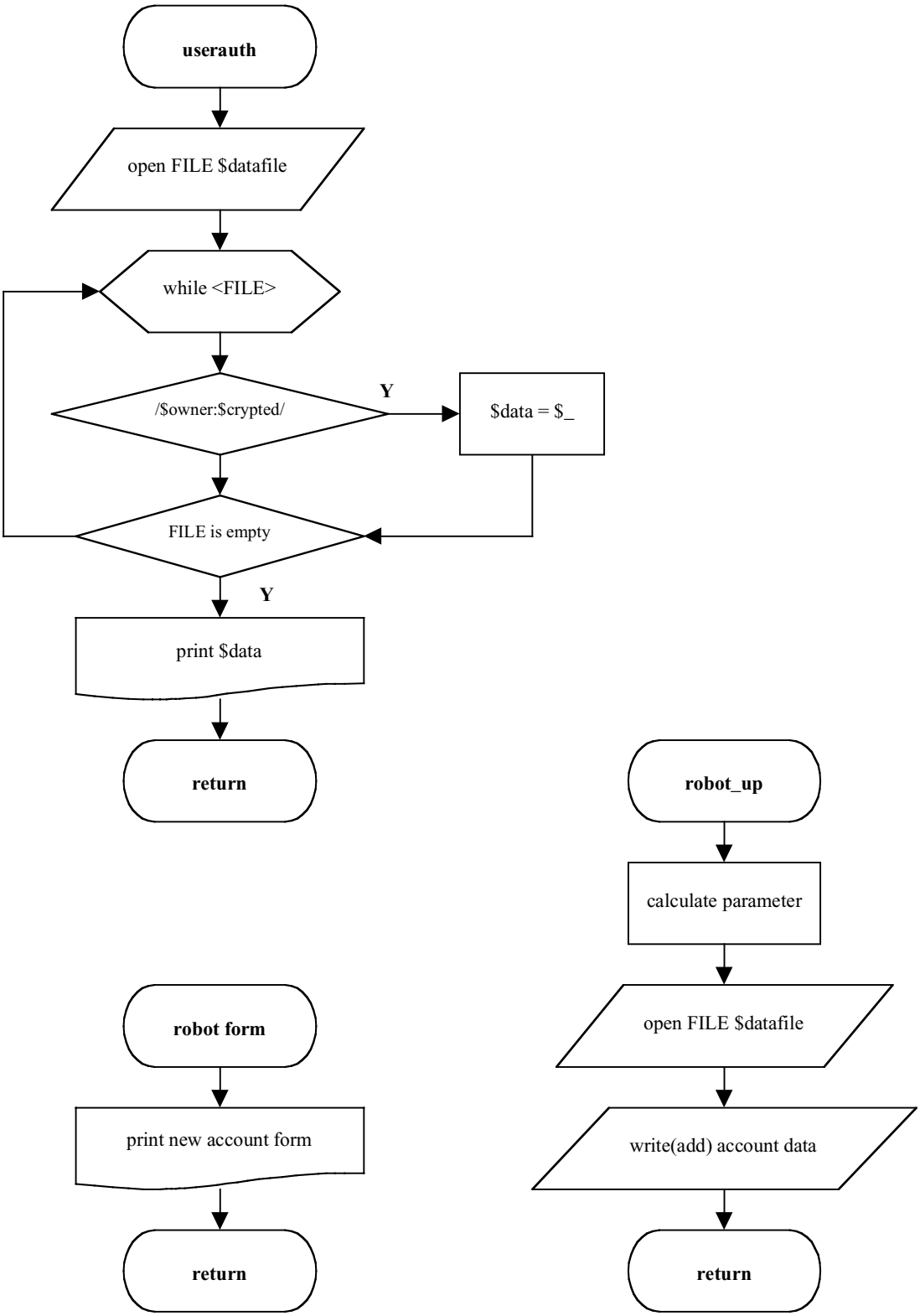
userauth
open FILE $datafile
while <FILE>
/$owner:$crypted/
Y
$data = $_
FILE is empty
Y
print $data
return
robot_up
calculate parameter
open FILE $datafile
write(add) account data
return
robot form
print new account form
return

図 4-d

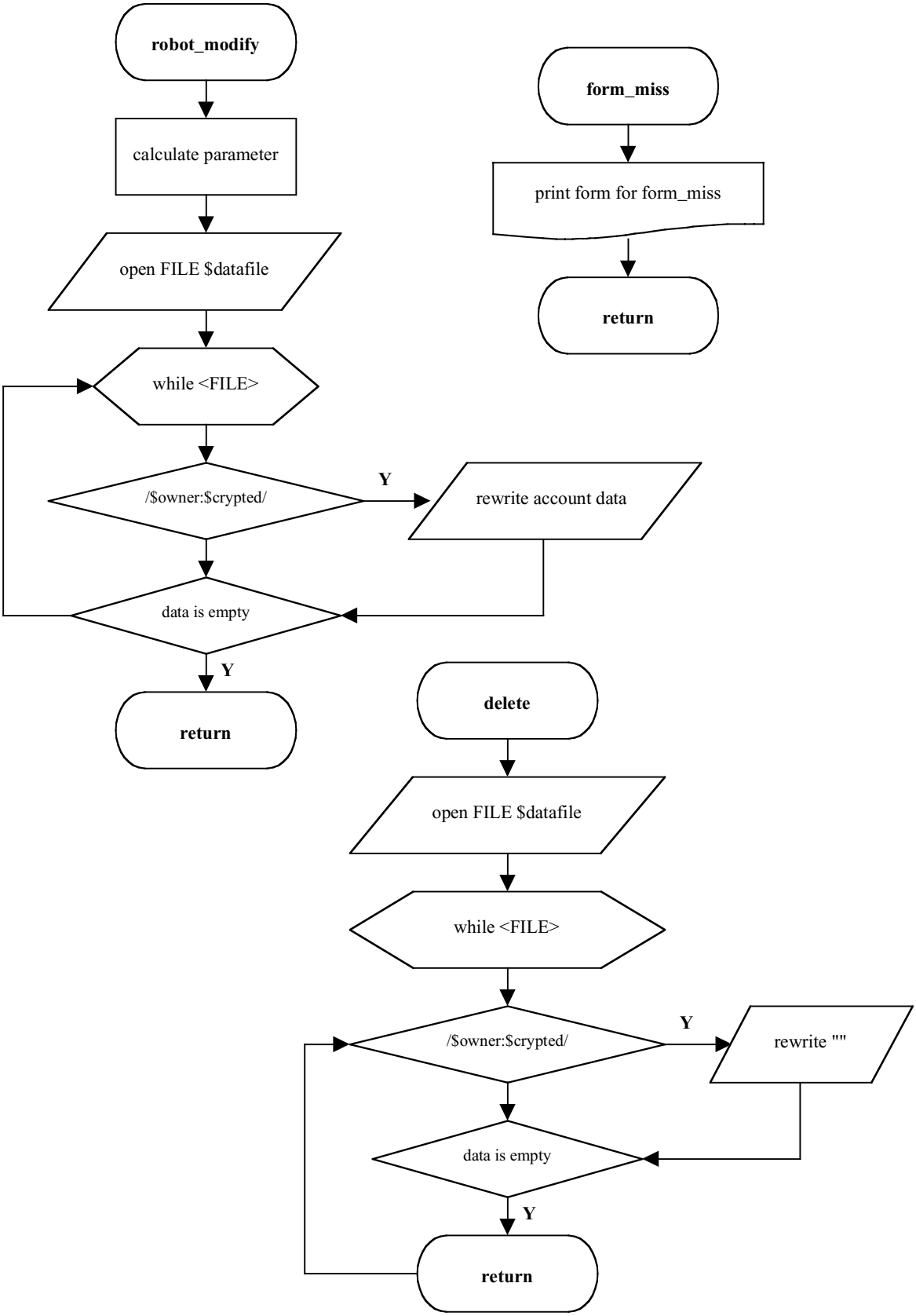
robot_modify
calculate parameter
open FILE $datafile
while <FILE>
/$owner:$crypted/
Y
rewrite account data
data is empty
Y
return
form_miss
print form for form_miss
return
delete
open FILE $datafile
while <FILE>
/$owner:$crypted/
Y
rewrite ""
data is empty
Y
return

図 4-e

3.2.2　アカウント操作プログラム

ゲーム内では「倉庫」と呼んでいる。この CGI はプレイヤの名前やロボットの名前、パーツ情報、評価値などプレイヤ情報の新規登録、変更、削除を行うためのプログラムである。プレイヤがまずここに最初に訪れた場合、ロボットを新たに登録してもらう。入力は HTML のフォームを武器ファイル、胴体部品ファイル、足部品ファイルからそれぞれ現状のデータを読み、CGI で入力フォームを動的に変化させる。このフォームを入力としてロボットのパラメータを計算し、プレイヤ情報をアカウントデータファイルに書き込む。入出力のフォーマットは、

"オーナー名:暗号化済パスワード:電脳兵名:武器名:胴体部品名:足部品名:武器攻撃力:攻撃硬直時間:基本行動硬直時間:防御力:装甲:武器命中率:武器品質:全体重量:武器射程:行動染色体の評価値<c>行動染色体<g>行動染色体の評価値<c>行動染色体<g>～(略)～<g>行動染色体の評価値<c>行動染色体· n"

のように一行に 1 人分のプレイヤの情報が保存されている。

①入力データ

<プレイヤのフォーム>

・オーナー名	$owner
・アクセスコード	$passwd
・電脳兵名	$robot
・武器	$weapon
・胴体部品	$body
・足部品	$leg
・人工知能が攻撃が成功したときの評価値の増量	$hit_succ
・人工知能が攻撃が失敗したときの評価値の減量	$hit_fail
・人工知能が回避が成功したときの評価値の増量	$avoid_succ
・人工知能が回避が失敗したときの評価値の減量	$avoid_fail
・人工知能が移動が成功したときの評価値の増量	$move_succ
・人工知能が移動が失敗したときの評価値の減量	$move_fail

<武器ファイル>

・武器名	$weapon_name
・武器攻撃力	$power
・攻撃硬直時間	$delay
・武器命中率	$hit
・武器品質	$qual
・重量	$weigh
・武器射程	$range

<胴体部品ファイル>

・胴体部品名	$body_name
・防御力	$armor
・装甲	$life
・重量	$weigh

<足部品ファイル>

・足部品名	$leg_name
・攻撃硬直時間	$delay
・基本行動硬直時間	$slow
・命中率補正	$hit
・重量	$weigh

②ロボットデータの計算式

・攻撃力	＝	武器の$power
・攻撃硬直時間	＝	武器の$delay+足部品の$delay
・基本行動硬直時間	＝	足部品の$slow
・防御力	＝	胴部品の$armor
・装甲	＝	胴体の$life
・武器命中率	＝	武器の$hit+足部品の$hit
・武器品質	＝	武器の$qual
・武器射程	＝	武器の$range
・全体重量	＝	武器の$weigh+胴部品の$weigh+足部品の$weigh

③アカウントデータファイルへの書き出し

・オーナー名	$owner
・アクセスコード	$passwd
・電脳兵名	$robot
・武器名	$weapon
・胴体部品名	$body
・足部品名	$leg
・武器攻撃力	$power
・攻撃時硬直時間	$delay
・基本行動硬直時間	$slow
・防御力	$armor
・装甲	$life
・武器命中率	$hit
・武器品質	$qual
・武器射程	$range
・全体重量重量	$weigh
・人工知能が攻撃が成功したときの評価値の増量	$hit_succ
・人工知能が攻撃が失敗したときの評価値の減量	$hit_fail
・人工知能が回避が成功したときの評価値の増量	$avoid_succ
・人工知能が回避が失敗したときの評価値の減量	$avoid_fail
・人工知能が移動が成功したときの評価値の増量	$move_succ
・人工知能が移動が失敗したときの評価値の減量	$move_fail
・勝敗数	$judg
・行動染色体列	$chromo_file

この CGI のフローチャートを以下の図 5-a～5-d に示す。

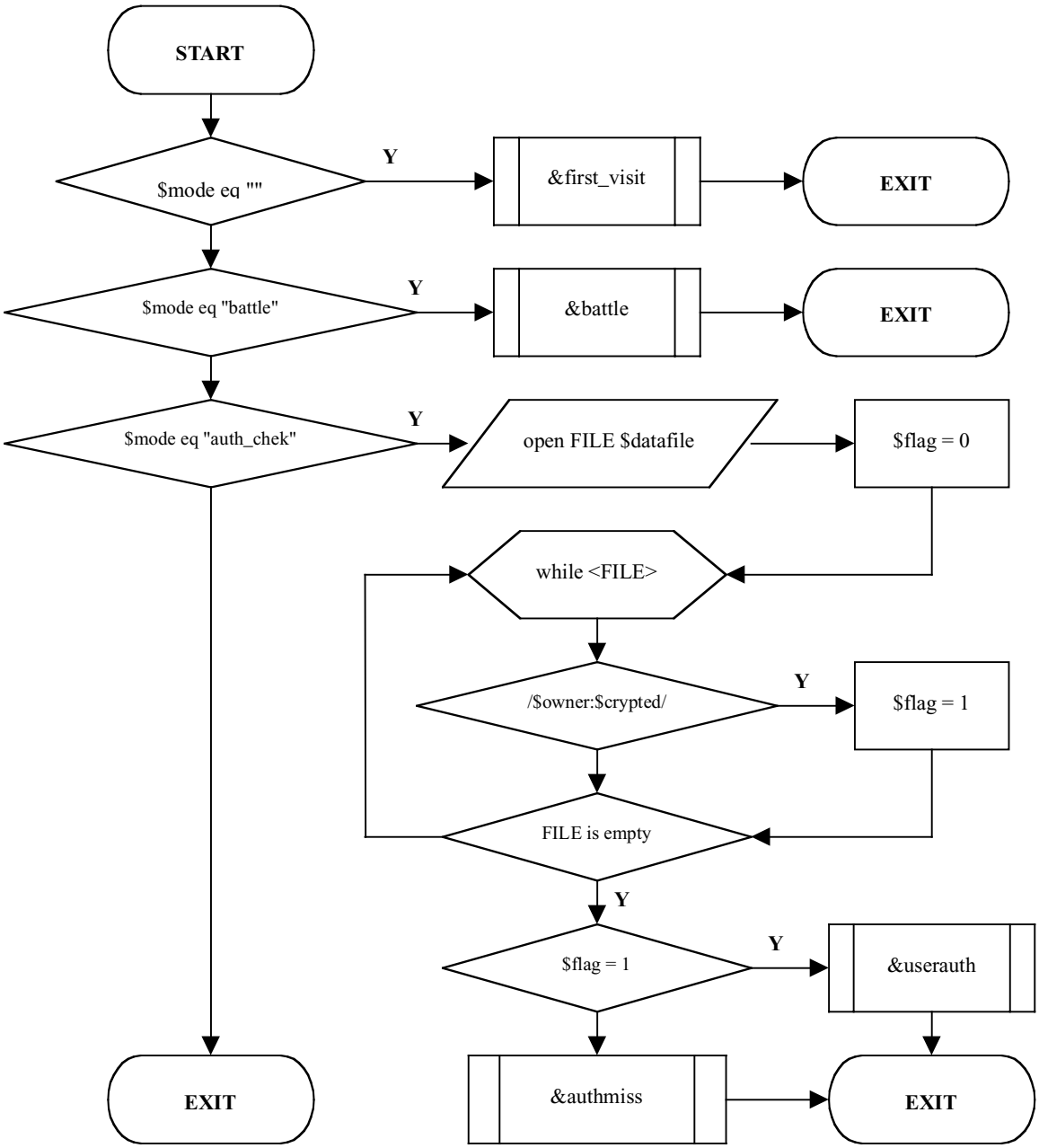
START
$mode eq ""
Y
&first_visit
EXIT
$mode eq "battle"
Y
&battle
EXIT
$mode eq "auth_chek"
Y
open FILE $datafile
$flag = 0
while <FILE>
/$owner:$crypted/
Y
$flag = 1
FILE is empty
Y
$flag = 1
Y
&userauth
EXIT
&authmiss
EXIT

図 5-a

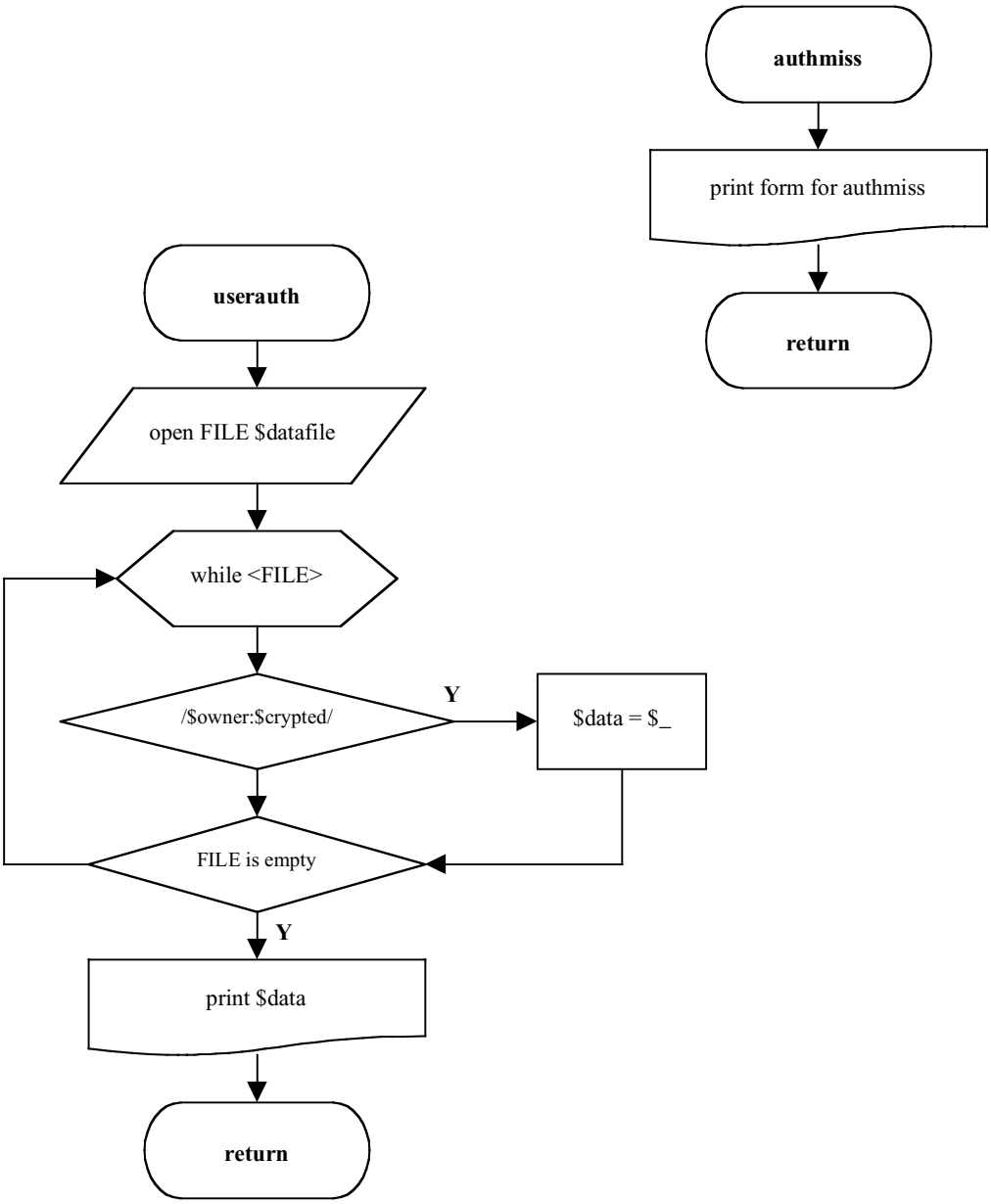
authmiss
print form for authmiss
return
userauth
open FILE $datafile
while <FILE>
/$owner:$crypted/
Y
$data = $_
FILE is empty
Y
print $data
return

図 5-b

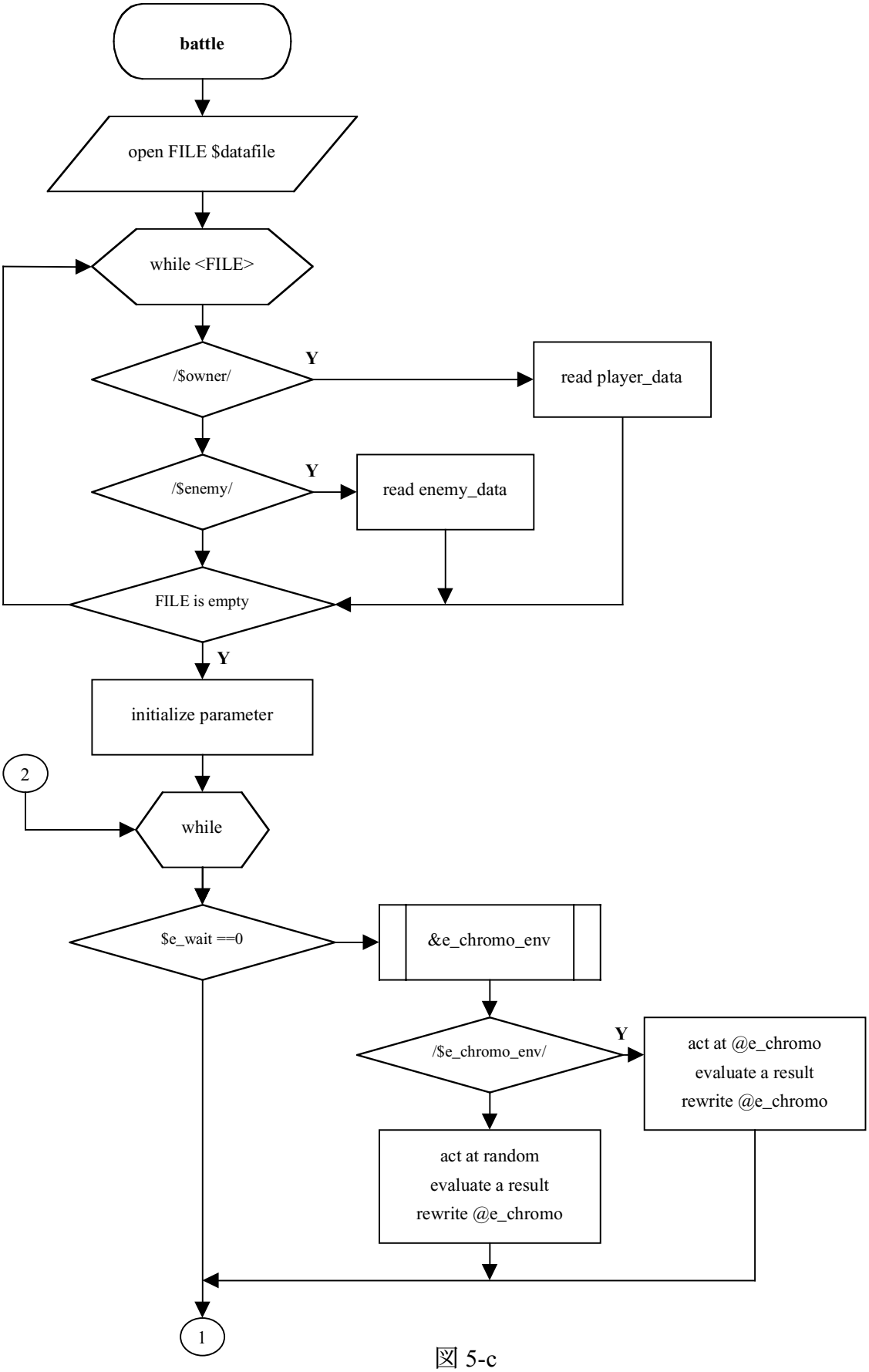
battle
open FILE $datafile
while <FILE>
/$owner/
Y
read player_data
/$enemy/
Y
read enemy_data
FILE is empty
Y
initialize parameter
2
while
$e_wait ==0
&e_chromo_env
/$e_chromo_env/
Y
act at @e_chromo
evaluate a result
rewrite @e_chromo
act at random
evaluate a result
rewrite @e_chromo
1

図 5-c

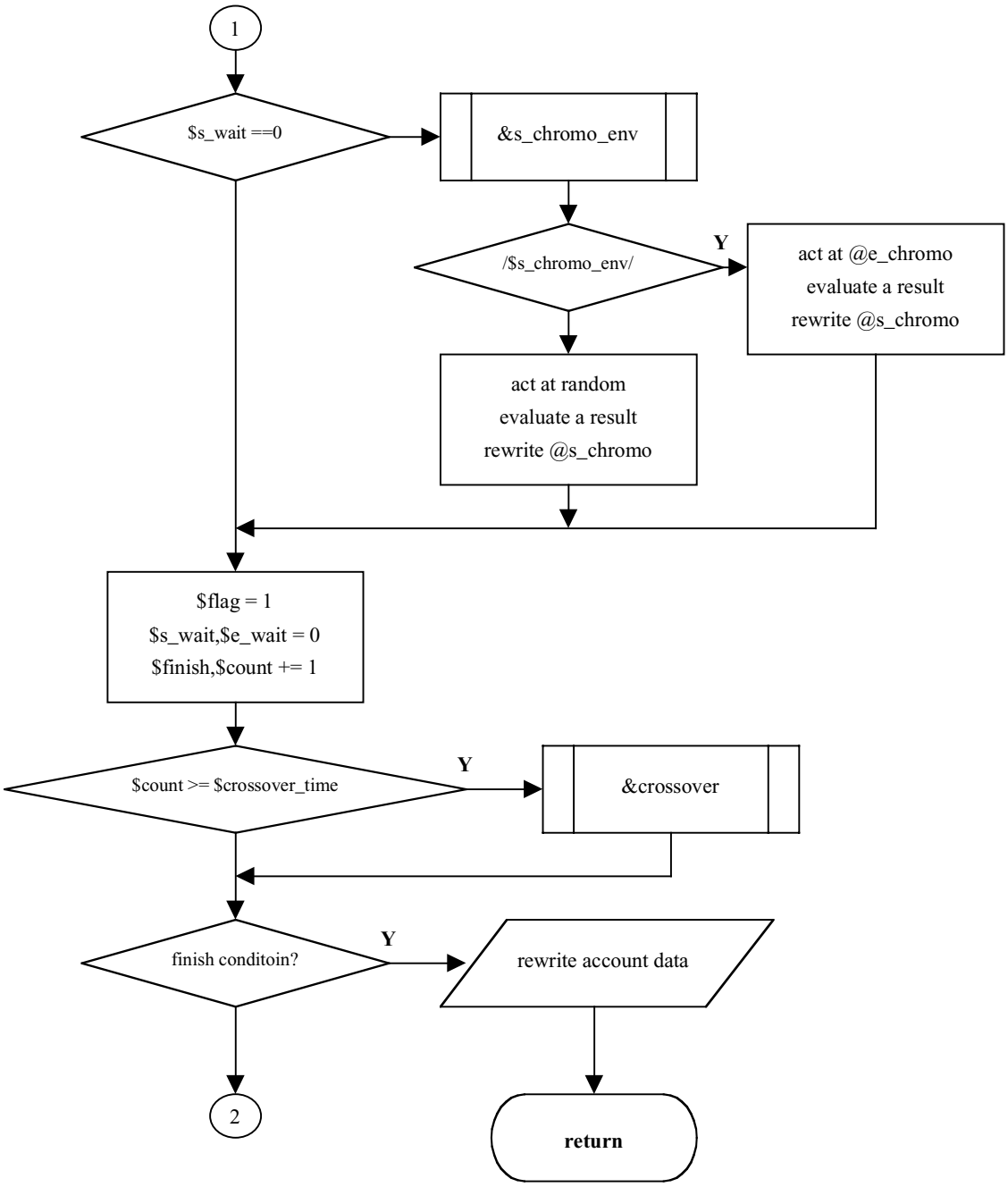
1
$s_wait ==0
&s_chromo_env
/$s_chromo_env/
Y
act at @e_chromo
evaluate a result
rewrite @s_chromo
act at random
evaluate a result
rewrite @s_chromo
$flag = 1
$s_wait,$e_wait = 0
$finish,$count += 1
$count >= $crossover_time
Y
&crossover
finish conditoin?
Y
rewrite account data
2
return

図 5-d

3.2.3　部品表示プログラム

この CGI は現在の部品状況がどのようになっているのか表示する CGI である。武器ファイル、胴体部品ファイル、足部品ファイルからデータを読み出し、表にして出力する。この CGI のフローチャートを図 6 に示す。

武器ファイルのフォーマットは、

"武器認識名:武器名:武器攻撃力:攻撃硬直時間:武器命中率:武器品質:重量:武器射程· n"

と 1 行に 1 つの武器データが格納されている。

胴体部品ファイルのフォーマットは、

"胴体部品認識名:胴体部品名:防御力:装甲:重量· n"

と 1 行に 1 つの胴体部品データが格納されている。

足部品ファイルのフォーマットは、

"足部品認識名:足部品名:攻撃硬直時間:基本行動硬直時間:重量· n"

と 1 行に 1 つの足部品データが格納されている。

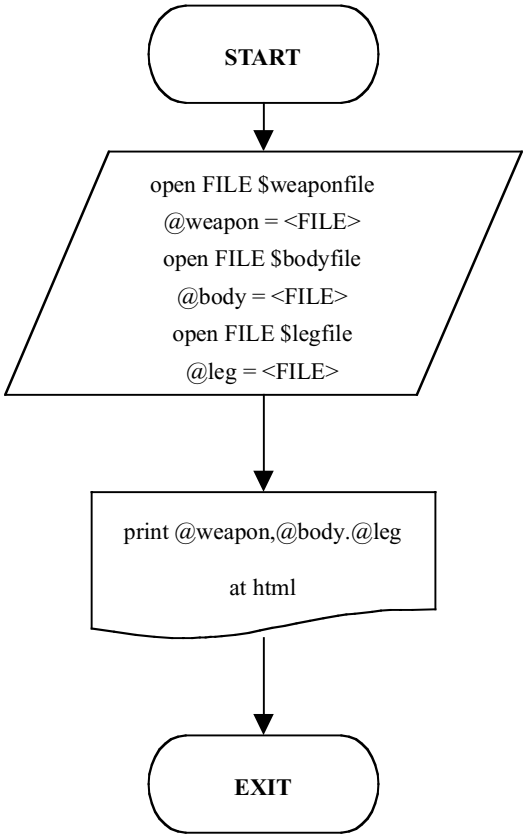

図 6

3.2.3　ランキング表示プログラム

この CGI はプレイヤのアカウントデータファイルからその時点での勝率を計算し、勝率の高い順に並び替え表にして出力するプログラムである。この CGI のフローチャートを図 7 に示す。

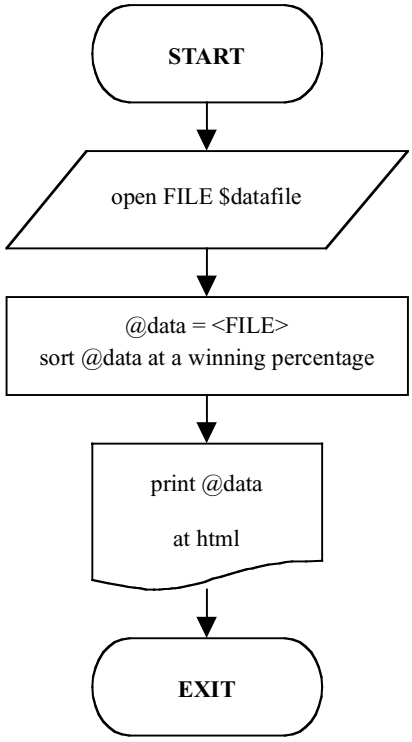

図 7

3.2.5　ウェブページ

戦闘処理プログラム、アカウント操作プログラム、商品表示プログラム、ランキング表示プログラムを案内するための、ウェブページを作成しなくてはならない。戦闘処理プログラムとアカウント操作プログラムは会員のみがアクセス可能にする。ウェブページ全体の構成を図 8 に示す。

```
/public_html┬battleartifact.htm
            ├top.htm
            ├news.htm
            ├menu.htm
            ├/robot_battle(for membership)─account_ctrl.htm
            └/cgi-bin┬/ba─flashbbs.cgi(free)
                     ├/lib─cgi-lib.pl
                     └/robot_battle┬robot_data_up.cgi(for membership)
                                   ├robot_battle_htm.cgi(for membership)
                                   ├parts.cgi
                                   ├sort.cgi
                                   ├account.dat
                                   ├weapon_data.dat
                                   ├body_data.dat
                                   └leg_data.dat
```

図 8　ウェブページの構成

3.3　実験

クラシファイアーシステムを実装した戦闘処理プログラムにおいて、挙動を調べるため、次のような実験を行った。

3.3.1　実験 1

全く性能の同じロボット A、B の 2 体を準備する。行動染色体はランダムに生成された初期の状態で、両者共に 30 個ずつ保持している。その状態から、①A,B 共に学習しないで戦闘、②A だけ学習しながら戦闘、③B だけ学習しながら戦闘、④A,B 共に学習しながら戦闘の 4 パターンについて調べる。回数は①、②、③、④それぞれ 500 回の戦闘を行った。ロボットの設定は以下の示す。

武器:ビームソード　胴部品:軽さ重視　足部品:二足歩行

攻撃成功評価値:10　攻撃失敗評価値:1　回避成功評価値:15

回避失敗評価値:3　移動成功評価値:2　移動失敗評価値:1

武器攻撃力:350　攻撃硬直時間:110　基本行動硬直時間:10　防御力:10

装甲:1000　命中率:55　武器品質:9　重量:160　武器射程:1

3.3.2　結果 1

実験の結果を表 3 に示す。

		B	
		学習なし	学習あり
A	学習なし	①A:138 勝 161 敗 201 分(勝率 32.20%) B:161 勝 138 敗 201 分(勝率 27.60%)	③A:10 勝 367 敗 123 分(勝率 2.00%) B:367 勝 10 敗 123 分(勝率 73.40%)
	学習あり	②A:357 勝 81 敗 62 分(勝率 71.40%) B:81 勝 357 敗 62 分(勝率 16.20%)	④A:152 勝 342 敗 6 分(勝率 30.40%) B:342 勝 152 敗 6 分(勝率 68.40%)

表 3　学習なしと学習ありの比較

3.3.3　考察 1

表 4 の①から、A、B 両方とも学習しないで戦闘を行う場合は、勝率は A、B 共に低く、戦闘回数のおよそ 40%が引き分けであった。これは行動染色体がランダムに生成された初期状態のままであるため、自分の行動ごとに評価されていない行動染色体の行動を選択するか、記述されていない場合、ランダムに行動をとるので、その行動の良し悪しがわかる指標をもっていない。さらに、行動に対しての評価がなされないため、ロボットの持つ行動染色体が進化しない。このために、初期の行動染色体が淘汰されずに残ったと考えられ、この行動染色体のために無駄な行動が多くなり、よって引き分けが 40%にも増加したと推測できる。

②の A のみ学習ありか、③の B のみ学習ありのどちらかでは、学習しない方に比べて②では 4 倍、③では 36 倍もの勝率の差を示した。これは、前述したように、学習しない方は無駄な行動染色体が多かったためと考えられ、反対に学習した方は有益な行動染色体を獲得できたと推測される。

④では A、B 共に学習させながら戦闘させたのに、勝率に大きな偏りが見られた。そこで、さらに戦闘を重ねた。1500 回まで戦闘を行わせた結果を表 4 に示す。

戦闘回数	A	B
250	80 勝 164 敗 6 分(勝率 32.00%)	164 勝 80 敗 6 分(勝率 65.60%)
500	152 勝 342 敗 6 分(勝率 30.40%)	342 勝 152 敗 6 分(勝率 68.40%)
750	349 勝 391 敗 10 分(勝率 46.47%)	391 勝 349 敗 10 分(勝率 52.06%)
1000	515 勝 468 敗 17 分(勝率 51.45%)	468 勝 515 敗 17 分(勝率 46.75%)
1250	621 勝 608 敗 21 分(勝率 49.64%)	608 勝 621 敗 21 分(勝率 48.60%)
1500	715 勝 750 敗 35 分(勝率 47.67%)	750 勝 715 敗 35 分(勝率 50.00%)

表 4　戦闘回数と成績

戦闘回数が 500 回程度では、まだ勝率に 2 倍以上の差が見られたが、1000 回で A は B の勝率を上回る結果となった。またその差もみられない。さらに、1500 回と戦闘を重ねるが、勝率がほぼ 50%付近に安定していることが確認できた。①の学習しない場合よりも勝率が高く、引き分けの数も 3%以下となった。しかしながら、学習の速度に差がでるのは、乱数を用いることによる統計的な偏りだと思われる。

以上をまとめると、学習した方が勝率が上がることがわかる。このことからクラシファイアーシステムを用いた学習は機能していると判断できる。さらに、このクラシファイアーシステムによる学習が効果的であるか、実験 2 で検証する。

3.3.4　実験 2

設定が違う A～D の 4 種類のロボットを用意する。1 ロボット当たり 500 回のリーグ戦を行い、計 1500 回の戦闘で学習させた後、その学習結果状態を固定してリーグ戦を行い、この時の結果を統計的に考察する。1 ロボット当たり 50 回のリーグ戦を計 150 回の戦闘を 1 セットとし、全部で 10 セット行う。4 種類のロボットの主な設定を表 5 に示す。なお、ここで設定した行動評価値はテスト段階時にテストロボットを相手にプレイした結果、いい成績をおさめたときの行動評価値である。

	A	B	C	D
武器	レーザー	ビームソード	ボム	ナパーム
胴体部品	バランス重視	軽さ重視	固さ重視	装甲重視
足部品	キャタピラ	二足歩行	八足歩行	四輪タイヤ
武器攻撃力	500	300	1000	450
攻撃硬直時間	100	150	130	120
基本行動硬直時間	50	20	30	20
防御力	40	30	90	50
装甲	1300	900	1200	1800
命中率	77	53	58	70
武器品質	8	7	2	7
全体重量	470	190	660	540
武器射程	3	1	10	8
攻撃成功評価値	10	10	10	10
攻撃失敗評価値	1	1	1	1
回避成功評価値	15	15	15	15
回避失敗評価値	3	3	3	3
移動成功評価値	1	1	1	1
移動失敗評価値	2	2	2	2

表 5　ロボットの設定

3.3.5　結果 2

リーグ戦の結果を表 6 に学習後の行動染色体の分布を表 7 に示す。

	A	B	C	D
1 セット目	60-48-42(40.00)	46-39-65(30.67)	38-72-40(25.33)	64-49-37(42.67)
2 セット目	52-63-35(34.67)	79-38-33(52.67)	62-69-19(41.33)	53-76-21(35.33)
3 セット目	53-69-28(35.33)	70-52-28(46.67)	44-80-26(29.33)	79-45-26(52.67)
4 セット目	49-71-30(32.67)	112-25-13(74.67)	47-96-7(31.33)	53-69-28(35.33)
5 セット目	47-73-30(31.33)	80-46-24(53.33)	59-75-16(39.33)	66-58-26(44.00)
6 セット目	44-73-33(29.33)	69-40-41(46.00)	65-60-25(43.33)	59-64-27(39.33)
7 セット目	45-74-31(30.00)	83-35-32(55.33)	57-74-19(38.00)	60-62-28(38.00)
8 セット目	61-65-24(40.67)	113-15-22(75.33)	26-109-15(17.33)	55-66-29(36.67)
9 セット目	35-86-29(23.33)	100-32-18(66.7)	61-70-19(40.67)	58-66-26(38.67)
10 セット目	46-73-31(30.67)	83-46-21(55.67)	64-68-18(42.67)	59-65-26(39.33)

表 6　リーグ戦の勝敗結果　　注) 勝数-負数-引分数(勝率%)

3.3.6　考察 2

表 6 から、ロボットの強さに差があるかどうか、勝率から推測してみる。まず、帰無仮説として「4 体のロボットの強さの差はない」とし、対立仮説として、「4 体のロボットの強さの差はある」とする。実験結果から、平均値と標準偏差を求めた値を表 7 に示す。

	A	B	C	D
平均値	32.80	55.67	34.87	40.02
標準偏差	4.85	12.97	8.24	4.95

表 7　平均値と標準偏差

表 7 から、一元配置分散分析の計算を行ったところ、表 8 のようになった。

要因	平方和	自由度	平均平方	F 値
群間	3206.13	3	1068.71	15.0446
群内	2557.29	36	71.0359	
全体	5763.42	39	147.78	

表 8　一元配置分散分析結果

F 分布表から F 値で有意差を見ると、1%有意水準でこの F 値が有意である、という結果が出た。すなわち、帰無仮説「4 体のロボットの強さの差はない」は棄却された。以上のことから、4 体のロボットの強さに差があるということに結論づけられる。それでは、この実験で最も勝率が高かったロボット B がゲーム中最強のロボットであり、他のどんなロボットでも勝てないのだろうか。しかし、初期設定で用いた行動評価値がロボット A～D まで同じ値に設定した点に疑問が残る。そこで、ロボットのタイプによって行動評価値を変え、学習をコントロールすべきではないかと考える。実験 3 ではロボットに学習させる過程のシミュレーション実験を行う。

3.3.7　実験 3

新たに設定した A ロボットと、前実験で使用した B ロボットを戦わせる実験を行う。プレイヤの操作を想定し、ロボットが強くなる過程をシミュレートする。初期設定は表 9 の通りである。A の設定は B と対称的なタイプとし、行動評価値は最初すべて 1 と設定した。この実験では、全く学習していない状態から、A、B 共に学習しながら 250 回の戦闘を行う。この戦闘後、成績が悪かった方をカスタマイズし、行動染色体を学習前の状態に戻した後、再度 250 回の戦闘を行う。以上を繰り返し成績がどのように変化するか時系列で観察する。

	A	B
武器	ボム	ビームソード
胴体部品	装甲重視	軽さ重視
足部品	キャタピラ	二足歩行
武器攻撃力	1000	300
攻撃硬直時間	130	150
基本行動硬直時間	50	20
防御力	50	30
装甲	1700	900
命中率	48	53
武器品質	2	7
全体重量	640	190
武器射程	10	1
攻撃成功評価値	1	1
攻撃失敗評価値	1	1
回避成功評価値	1	1
回避失敗評価値	1	1
移動成功評価値	1	1
移動失敗評価値	1	1

表 9　ロボットの設定

3.3.8　結果 3

①250 回の戦闘　A:49 勝 108 敗 93 分　B:108 勝 49 敗 93 分

②A:攻撃の頻度を上げるため、攻撃成功評価値を 1→10

③250 回の戦闘　A:53 勝 153 敗 44 分　B:153 勝 53 敗 44 分

④A:移動もしないと射程に捕えられないため、移動成功評価値を 1→5

⑤250 回の戦闘　A:101 勝 67 敗 82 分　B:67 勝 101 敗 82 分

⑥B:攻撃の頻度が低いので、攻撃成功評価値を 1→10

⑦250 回の戦闘　A:26 勝 214 敗 10 分　B:214 勝 26 敗 10 分

⑧A:相手の方が素早いので間合いを同じにして打ち合う。武器をボム→ビームソード

⑨250 回の戦闘　A:24 勝 170 敗 56 分　B:170 勝 24 敗 56 分

⑩A:さらに攻撃の頻度を上げるため、攻撃成功評価値を 10→20

⑪250 回の戦闘　A:111 勝 61 敗 78 分　B:61 勝 111 敗 78 分
⑫B:回避を多様するため、回避成功評価値を 1→15
⑬250 回の戦闘　A:102 勝 87 敗 61 分　B:87 勝 102 敗 61 分
⑭B:無駄な動きがあるかもしれないので、移動失敗評価値を 1→2
回避失敗評価値を 1→3
⑮250 回の戦闘　A:124 勝 98 敗 28 分　B:98 勝 124 敗 28 分
以上のような戦闘の流れであった。最後の戦闘が終わった時点のロボットの設定を表 10 に示す。

	A	B
武器	ボム→ビームソード	ビームソード
胴体部品	装甲重視	軽さ重視
足部品	キャタピラ	二足歩行
武器攻撃力	1000→300	300
攻撃硬直時間	130→90	150
基本行動硬直時間	50	20
防御力	50	30
装甲	1700	900
命中率	48→63	53
武器品質	2→7	7
全体重量	640→520	190
武器射程	10→1	1
攻撃成功評価値	1→20	1→10
攻撃失敗評価値	1	1
回避成功評価値	1	1→15
回避失敗評価値	1	1→3
移動成功評価値	1→5	1
移動失敗評価値	1	1→2

表 10　ロボットの設定の変更後

3.3.9　考察 3

時系列のシミュレーションから、少し設定を変えただけで、勝率がかなり変化することが確認できた。実際に戦闘と設定を繰り返す場合には、相手の弱点をつく必要がある。また前実験でもっとも勝率が高かったロボット B は最終段階⑮の戦闘の

結果、ロボット A よりも勝率が低いことが確認された。これはロボットがどんなロボットにも勝てるような完全な設定は存在しないことを示している。さらに、B の武器をビームソードからボムに変更し、250 回戦闘をさせたところ以下の結果が出た。
⑯B:相手の武器射程からを遠ざかるため、武器をビームソード→ボム
⑰250 回の戦闘　A:11 勝 236 敗 3 分　B:236 勝 11 敗 3 分
このように、軽くて装甲が薄いロボットは相手の武器射程から離れるように、移動し、適宜攻撃することで勝利することができ、また反対に重いが装甲が厚いロボットは、相手との武器射程を合わせ、打ち合うような戦いをすることで、勝利をおさめられる。このようなシステムを実現することで最強設定を回避できるのと考える。

3.4　まとめ

本研究で構築したゲームシステムについて概説した。その中でも特に重要である、戦闘でのクラシファイアーシステムによる学習は有効であることを示した。また、実験から最強設定の回避方法を示した。最強設定を回避できると、駆け引きがより面白くなる。

第　4　章
お わ り に

4.1　結論

目的であった、

- 総合ゲームサイト(web ページ)の
- クラシファイアーシステムを実装したロボット作成プログラムと戦闘処理プログラムによるテキストタイプの戦闘結果出力
- 個人ランキング表示
- 電子掲示板
- 十分なバランス調整
- インターネットブラウザのみでゲームをプレイできる
- 発展ゲーム
- 非同期システム

これらのゲームシステムは実装できた。

実験から、戦闘でクラシファイアーシステムによる学習が有効であることを示した。このシステムは既存のゲームになかったシステムであるので、新規性があると考えられる。同じロボット同士の戦闘でも、学習の速度が異なっていることや、プレイヤが諸パラメータを設定できることから、意外性は十分にあると考える。武器や、胴体部品、足部品などの性能を調整し、どのようなタイプのロボットでも設定次第で勝てるように、良好なゲームバランスになったと考える。最強設定の回避と、CGI でのランキング制を実現し、競争の概念を満たす。さらに、電子掲示板によりコミュニケーションをとることができる。

以上のことから、面白さの 5 要素、
・新規性
・ゲームバランス
・意外性
・競争の概念
・コミュニケーション
は実現した。またネットワークゲームという供給形態を採用し、ゲームに組込んだ。従って、本研究で作成したゲームは「消費」されないゲームであると推定できる。

4.2　今後の展望

一般公開し、アンケートなどを取ることで、ゲームが面白かったか検証する必要がある。

このゲームのストーリーやキャラクタを専門の作家やデザイナーに制作してもらうことにより、ストーリー性やキャラクタ性を満たすことができると考える。また、テキストベースの戦闘ではなく、グラフィカルで、ポリゴンを使用した 3D グラフィック処理などが可能になると、エンタテイメント性の実現も可能である。

現システム内でも、新しい武器や胴体部品、足部品などを、プレイヤがある一定条件の下で作れるようにする。市場には一定数しか存在できないことにし、人気のない部品に新製品が取って代わるプログラムを用意することで、競争の概念がそこに生まれ、新製品に関するディスカッションや討論が行われると考える。それらがさらにゲームをより面白くし、長くゲームを遊んでもらうことにつながる。

今後、ゲームは作り手からプレイヤに一方的に提供されるものではなくなるだろう。ゲームは作り手とプレイヤとの双方向の情報交換などの相互作用により、発展していくものである。それには、固定ゲームではなく、発展ゲームでなくてはならない。また、コンピュータと対戦するよりも、人間との対戦の方がゲームはより面白くなる。これから先、人と人とのコミュニケーションを重要視したゲームを望む。

謝 辞

本研究を進めるにあたり，研究テーマの決定から、研究の進め方、その他さまざまな面で指導して下さった櫻井彰人教授に深く感謝いたします。また，人工知能に関する問題や論文作成において適切なご助言をいただいた荒木修助手に感謝いたします。さらに、同研究室において，研究に関する助言や有用な情報をくださった北野陽一氏、鈴木昭吾氏をはじめ、日頃お世話になった櫻井研究室の皆様に感謝の意を表します。

参 考 文 献

[1] ロジェ・カイヨワ著, 多田道太郎・塚崎幹夫訳, 遊びと人間, pp.42-81, 講談社学術文庫, 1990.

[2] ロジェ・カイヨワ著, 多田道太郎・塚崎幹夫訳, 遊びと人間, pp.82-88, 講談社学術文庫, 1990.

[3] 平林久和, 赤尾晃一, ゲームの大學, MEDIA FACTORY, 1996.

[4] 長久勝, NPC 動作アルゴリズムの自動生成に関する考察, 情報処理, Vol.98, No.27, pp.55-60, 1998.

[5] 平野広美, 応用事例でわかる遺伝的アルゴリズムプログラミング, パーソナルメディア, 1995.

[6] 合原一幸, 複雑形がひらく世界, 日経サイエンス社, 1997.

[7] 佐野千遥, 人工知能と人工生命, 日刊工業株式会社, 1994.

[8] 上田完次, 下原勝憲, 伊庭斉志, 人工生命の方法, 工業調査会, 1995

[9] 杉山雄一, 和田理彦, ミリオンセラーを目指したアクションパズルゲーム XI[sai] の製作, bit, Vol.31, No.7, pp28-36, 共立出版, 1999.

[10] 深沢千尋, すぐわかる Perl, 技術評論社, 1999.

[11] 宮坂雅輝, CGI/Perl ハンドブック, SOFT BANK, 1999.

[12] 結城浩, Perl で作る CGI 入門基礎編, SOFT BANK, 1998.

[13] 千葉毅, HTML3.2 スーパーリファレンス, 株式会社ソーテック社, 1997