“Packetris”について

Team KOBASEMI

2012 年 5 月 18 日

目次

1　“Packetris”についての概要

本コンテストの課題は「パケットアート」である。パケットに含まれる情報を用いて何らかの「アート」を表現することが要求されている。

本作品の作成にあたって、「芸術とは何か」という問いに対して議論した結果、芸術とは絵であったり音楽であったり、文章であったり、動きだったりとさまざまな解を得ることができた。それをコンピュータ上のソフトウェアで実現し、芸術として表現するにはどういったものが最も適しているかを検討した結果、絵や音声や動きが複雑に絡み合い、コンピュータ上でしか実現できない「ゲーム」という表現媒体に着目した。そのうえで、パケットを 1 つの表現要素とし、パケットは通常複数個あることから、複数個で 1 つの要素を構成するものは何かということにたどりついた。そこで、複数個で 1 つの要素となるものとなるゲームの代表的なものとして「テトリス」や「ぷよぷよ」などのいわゆる「落ちものパズルゲーム」がある。このゲームの「テトリミノ」や「組ぷよ」などのような複数要素を含んだブロックをパケット情報をもとにして生成することでゲームとしての面白さを表現できる。

ゲームのルールについて、だれもが知っており、だれもが操作可能なものとして「テトリス」をベースとした作品を作ることになった。また、ひとつひとつにパケット情報を付与することで絵としての芸術性が見いだせるだろうと考えた。さらに、本コンテストの作品例に「VOCALOID をパケットの情報を用いて歌わせる」というものがある。そこで、音楽をパケット情報に基づいて自動的に生成することができるだろうという助言とその実験が行われ、本作品上の一部の音楽はパケットによる自動作曲が実装されている。

図 1　本作品のロゴマーク

「パケリス」という名前は、本コンテストの題材であり、ミノや音楽の生成材料となる「パケット」と、世界的にポピュラーな落ちものパズルゲームである「テトリス」を組み合わせた造語である。また、「パケット」すなわち”Packet”は「小包」と訳されることがあり、パケットを絵にする際によく矩形で描かれる。テトリスは複数の矩形のブロックを合わせて「ミノ」を表現しているため親和性が高い。

以下、”Packetris” と「パケリス」が混在して表記されるが、どちらも同じものを指す言葉である。

2 “Packetris” のインストール方法および確認済み動作環境

パケリスの実行環境は以下のようになる。
パケリスの実行確認済みの動作環境は以下のようになる。

- Windows XP、7、8 の各 32、64bit 版
- JRE(Java Runtime Environment) 7[*1]以上
- WinPcap[*2]もしくは、`libpcap.so`
- キーボードとマウスなどの入力装置
- 横 600 ピクセル、縦 800 ピクセル以上の画面
- 1GHz 以上の CPU
- 1GB 以上のメモリ

パケットキャプチャに用いているライブラリである JnetPcap[1] は、WinPcap もしくは、`libpcap.so` を必要とするため、事前のインストールが必要である。それ以外の環境での動作については保障しかねる。

この作品は依存する JnetPcap のライブラリが 32bit 版と 64bit 版に分かれているため、実行する環境のアーキテクチャに合わせたものをダウンロードする。ダウンロードした zip ファイルを解凍し、`Packetris.jar` をダブルクリック、もしくは

```
$ java -jar Packetris.jar
```

と入力し、実行する。なお、`java` へのパスは通っているものとし、”$”は入力プロンプトとする。

この作品をアンインストールする際は、解凍したフォルダごと消去すればよい。この作品はレジストリやその他のシステムに関わる設定ファイルに対して書き込みは行っていない。

3 パケットの使用用途について

本作品中で、パケットは主に 3 通りの使われ方をしている。

一つはミノの生成にパケットのプロトコルを使用している。ミノの形状をパケットによって生成することはミノの出現パターンが偏ってしまう可能性があるが、パケットのプロトコルを用いることでその通信内容がミノの形状として可視化されるというメリットがある。たとえば、Skype での通話が多い時の通信内容では UDP プロトコルが多く流れているため、その特徴をゲーム中にわかる仕組みになっている。

二つ目はミノの色である。これはパケットの様々な情報を複合的に組み合わせて実現している。そのため、キャプチャした結果を読み込ませることによって毎回違った色で操作でき、その色合い

[*1] Java.com `http://java.com/ja/`
[*2] WinPcap:`http://www.winpcap.org/`

を楽しむことができる。逆に同じキャプチャファイルを使用することで、ほぼ同じ生成パターンでゲームをプレイすることができるため、通常のテトリスとは違った楽しみ方もできる。

一つ目と二つ目を組み合わせることによって、パケットの情報可視化という点で面白い作品を作ることができたのではないかと考える。

三つめは音楽の自動作曲での使用である。これは音楽をパケットに含まれる IPv4 アドレスから作成している。ここでは、パケットから生成される音楽が不協和音的な音楽にならないように音楽理論に基づいた処理を行い、また、ある一定の縛りを設けることによって、耳に心地いい音楽を自動的に生成している。この音楽はパケリスのプレイ画面中で再生される。音楽によって、情報可視化ならぬ情報可聴化が可能になった。

4　パケット解析機について

本作品を作成するにあたり、パケットをキャプチャし、そのパケットを解析する必要がある。そのため、本作品作成の早い段階から、以下に挙げる特徴を持つパケット解析機を開発した。

4.1　あたらしいパケットの保持

制限付きのキューを作成し、到着したパケットを次々とキューに装填することで、新鮮なパケットを使うようにしている。

4.2　完全なイベントドリブンへの拡張

パケットの到着とそのプロトコルに即座に反応する独自のハンドラを用意し、「デバイスからロード」を行った場合、完全にリアルタイムなプロトコルデータの抽出を、 簡単に実装できるようにした。これはパケットアート以外にも流用可能である。

4.3　省メモリ性

パケットを一時的に保持することはメモリの浪費に直結するが、JNI(Java Native Interface) のライブラリが保持するオンメモリのデータへのポインタを一つしか持ち得ないクラスを作成し、使いまわすことでメモリの浪費を避けている。

4.4　データに特化した開発

WIDE プロジェクトでは L3 以上のレイヤがデータの肝となっているが、肝心のペイロードが除去されている。よって、必要以上の機能をつけることはやめ、L3、L4 を中心にした開発をした。

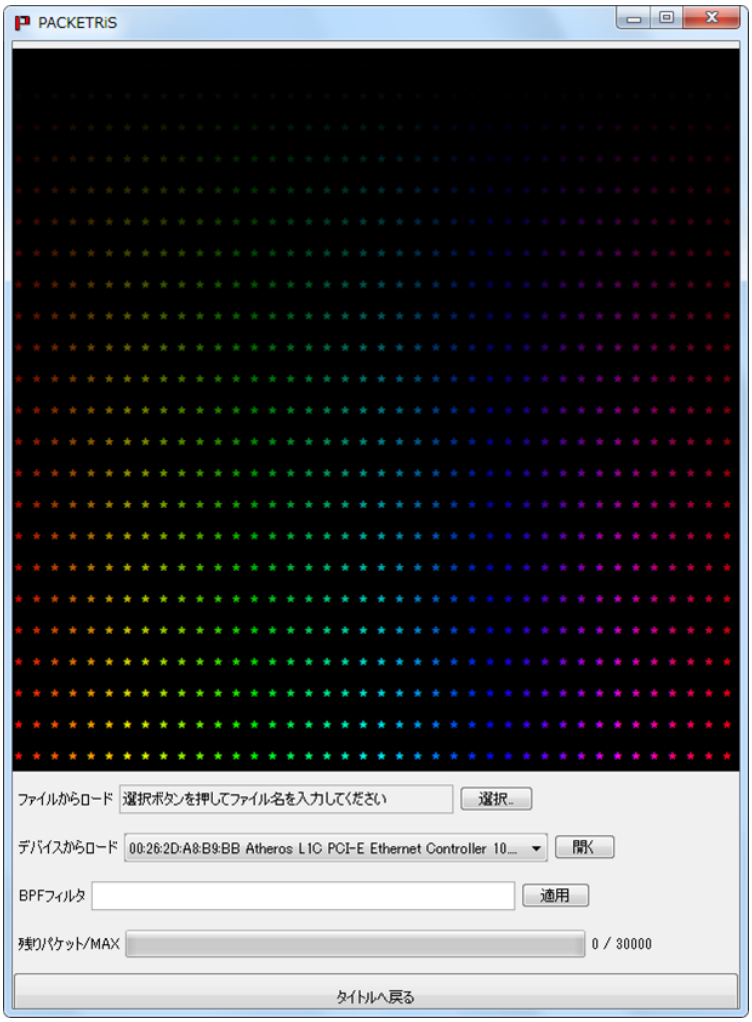

図 2　パケット解析機

5　ミノの生成方法について

本作品はミノをパケットから生成している。その評価方法について以下に説明する。

5.1　パケットの評価方法

ミノの形状を生成するにあたり、まずパケットを一個、ロードする。このパケットの読み込みは、デバイスまたはファイルからロードできる。パケット不足を極力防ぐため、ある程度の数のパケットをプログラムの中で保持している。次に、以下の表に基づいた情報を利用し、ミノを生成する。

このように複雑なのは、テトリミノの形が偏りすぎるのを防ぐためである。

次に、テトリミノ (4 つのブロックが組となっているミノ) とペントミノ (5 つのブロックが組となっているミノ) の両方がゲームに登場する設定の場合は IPv4 パケットを含むならばテトリミノ

表 1　ミノの形状の決定表

IPv4	TCP	シーケンス番号とチェックサム値
	UDP	送信元ポート番号と宛先ポート番号
	その他	IPID シーケンス番号とチェックサム値
IPv6	TCP	シーケンス番号とフラグフィールド
	UDP	チェックサム値と宛先ポート番号
	その他	フローラベルとデータペイロード長
その他		パケットサイズとミリ秒単位の時間

を、それ以外はペントミノを テトリミノのブロックの数の決定因子にしている。これも同様にバランスを考えてこのような条件にしている。

6　ミノの描画方法について

本作品は各ブロックに割り振られたパケットの情報を用いてミノのグラフィックを描画している。描画がどのように行われているかについて、以下で説明する。

ミノの彩色においては、Java における HSB 表色系の `Color(H,S,B)` を使用する。HSB 表色系とは、色を色相 (Hue)、彩度 (Saturation)、輝度 (Brightness) の三要素で表現する表色系である。一般にコンピュータ上で用いられている RGB よりも「鮮やかな暗い赤」のように人間にとって直観的な色の表現が可能になる。色相は一般的に円で表現され、色相環とも呼ばれる。

色相は変動で、ミノの持つパケットのチェックサムなどのパケットサイズを基軸からの回転量にした。ミノ毎に色分けをするためである。紛らわしい色の重複を避けている。彩度は固定で、1.0 にした。これは鮮やかさを高くすることでミノの視認性を高くする目的である。。輝度は変動で、ミノの持つパケットの最上階のレイヤが、OSI 参照モデルにおいて、低いほど暗い色になる。

本作品ではオプションからミノに対して特定の文字を表示できる機能がある。この文字を決定する方法として以下の表を用いる。この表は、ミノの描画に用いるパケットが”TCP over IPv4” のとき、「黄色の T」が表示される、と読む。

表 2　色の決定テーブル

	TCP	UDP	ICMP	IPv4	IPv6
IPv4	黄色の T	緑の U	シアンの I	マゼンタの 4	赤の 6
IPv6	青の T	青の U	白の I	青の 4	青の 6
Ethernet	赤の T	赤の U	白の I	白の 4	白の 6
PPP	赤の T	赤の U	白の I	白の 4	白の 6
その他	赤の T	赤の U	白の I	白の 4	白の 6

プロトコル名	色と文字
L2TP	朱の L
PPP	朱の P
ARP	朱の A
Ethernet	朱の E
OTHER	朱の!

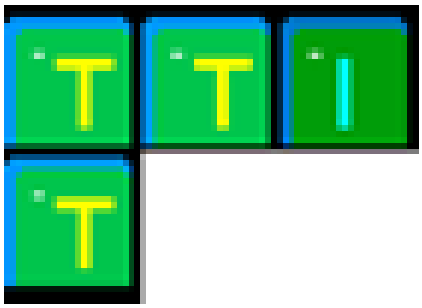

図 3　生成されるミノの例

7　パケット情報からの自動作曲について

本作品はパケットから音楽を自動で生成する機能が備わっており、パケットから自動生成された音楽を BGM にして、ゲームをプレイすることができる。音楽を自動生成する場合、ただただ音符を無造作に並べただけではただの不協和音になる。そこで、音楽理論に基づいた処理を行い、耳の心地よい音楽を生成するようになっている。

7.1　自動作曲法について

主旋律は IPv4 の値から自動生成している。IPv4 の 1 オクテットから 1/2 小節を生成する。伴奏は明暗 2 通りから固定で生成する。コード進行は固定である。主旋律は自動生成とは言え、伴奏と不協和音にならない程度に範囲を縛っている。ただ単にパケットの数字を音高に変換したものは、不協和音となってしまう可能性が非常に高く、音楽とは呼べない。

一般的に音楽は芸術的感性が重視されるように思われているかもしれないが、一種の規則性を持っており、それらは数学的に表すことができる。つまり、パケットのもつ数字などから協和音的な音楽を生成することは可能である。通常音楽にはコード進行があり、そのコード 1 つ 1 つにはそれぞれ適する音がいくつか存在する。そこで予めコード進行は決めておき、そのコードに合う音をパケットの数字を基にして当てはめる。

これにより使用するパケットによって毎回違う音楽が生成される。

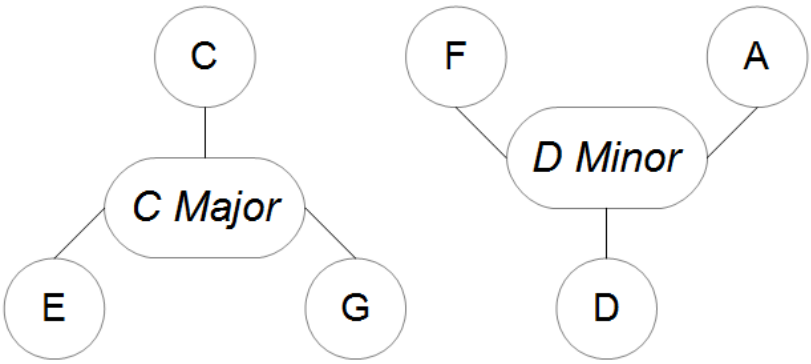

図 4　音楽の自動生成に用いるコード

7.2　音楽の自動生成に関わるクラスについて

音楽の自動生成に関わるクラスとその役割について以下に示す。

- CodeMaker
  BGM のコード進行を定義するクラスである。後々の BGM 生成に大きな影響を与える。このコード進行は自動生成ではなく、予め定義されたものとなっている。
- ScaleMaker
  BGM 生成に使用する鍵盤を定義するクラスである。不協和音を生成しないためには、正しい音階を用いて音楽を生成する必要がある。鍵盤には幹音 (白鍵) と派生音 (黒鍵) が同時に存在し、Java では低音から両者関係無く数字が順に割り振られている。長調と短調では一定の規則性で幹音と派生音の順番が決まっており、その規則性に従って使用する鍵盤を定義している。
- MelodyAlgorithm
  ロードしたパケット中から IPv4 を取り出して処理し、主旋律の決定要素を作成するクラスである。IPv4 アドレスをオクテット毎に取り出し、その数値の大きさによって判定要素 A を確定させている。また、ScaleMaker で定義したコードの協和音となる任意の音を定義し、これを判定要素 B としている。以上の判定要素 A と判定要素 B を組み合わせて算出されたものが、音楽の主旋律の決定要素となる。
- MelodyMaker
  BGM の主旋律を作成するクラスである。BGM の 1 小節は 4 拍で構成されており、うち 2 拍を MelodyAlgorithm の主旋律の決定要素 1 つを用いて生成する。ただ、それだけだと単調な音楽になってしまうため、偶数拍は先の決定要素の音を、奇数拍はそれを変化させた音を使用する。この部分がパケットを基にした BGM の自動生成要素となっている。
- AccompanimentMaker
  BGM の伴奏を生成するクラスである。伴奏は自動生成ではなく、ScaleMaker で定義した

コード進行に合う伴奏が予め定義されている。

- MusicPlayer
  自動生成した BGM を再生するクラスである。MelodyMaker と AccompanimentMaker で生成された BGM データ (sequence) を組み合わせ、再生する。
- MidiPlayer
  外部 Midi ファイルを読み込み、再生するクラスである。Midi ファイルを一度開き、音量情報を指定のものに全て書き換え、出力再生する。

8 システムについて

本作品は Java によって構成されている。特に、Java の AWT と Swing を活用して、複数の画面を制御している。画面の制御などに関してはパケットに関する処理を行っていないため、詳細は割愛する。本作品のソースコードは、Github[*3]で共有されている。また、本作品のソースコードは MIT License とする。

本作品は起動時に複数のファイル読み込みが発生するため、その読み込み待ち時間にスプラッシュスクリーンを表示するようにしている。そうすることでこの作品に触れる人が、ファイルの待ち時間に感じるストレスを小さくする狙いがある。

本作品はパケットからミノを生成するが、テトリミノの場合、7 種類しか存在しない。そのため、ペントミノを生成できるようにし、12 種類のミノを生成できるようにした。また、テトリミノとペントミノを同時に出現できるようにすることで、19 種類のミノを発生できるようにする。そのため、難易度はテトリミノのみ、ペントミノのみ、テトリミノとペントミノの混合と 3 種類に分けることができる。

9 本作品の開発手法

本作品は 1 月ごろに作品の草案となる「パケット」と「テトリス」の組み合わせについて議論した。本作品のプログラムの開発には、Processing[*4]や C++、Java が候補に挙がった。その中でプログラムの開発においてメモリリークなどの追跡困難なバグが発生しにくく、高い拡張性と、優秀なパケット読み込みライブラリが存在しており、開発コストと学習コストを鑑みた結果、Java を用いることになった。Java を用いるうえで、パケットキャプチャリングを行うためのライブラリとして JnetPcap を使用している。

本作品の開発に着手した時期は 2 月である。2 月は大学の春季休業期間中であるため、開発メンバーで集まって開発することは難しいと判断したため、Github を用いてソースコードの共有を行った。本システムの設計には UML(Unified Modeling Language) のクラス図やシーケンス図

[*3] Github:http://github.com/kobasemi/PacketArt
[*4] Processing:http://processing.org/

を用いており、オブジェクト指向の強いプログラムになっている。

本作品は vim、Eclipse、IntelliJ IDEA などのいろいろな IDE(Integrated Development Environment) や、Windows、Linux、Mac などの多様な環境での開発が行われていた。

参考文献

[1] JnetPcap Documentation，http://jnetpcap.com/jnetpcap-1.3，2013 年 5 月 18 日確認