1. 整体架构



1. 接收模块

1 GNURadio

gnuradio是软件无线电开源开发框架，用c++和python开发，支持USRP设备，该模块提供构建Iridium突发检测器和解调器的模块，可用于检测和解调来自 Iridium 卫星网络的数据。

1.1 先决条件

需要使用以下组件进行有效的[GNURadio]安装：

- VOLK  
 - FFT  
 - Filter  
 - Python  
 - SWIG

volk的全名是vector optimized library of kernel，是矢量优化库。  
 FFT是一种DFT的高效算法，称为快速傅立叶变换（fast Fourier transform）。  
 SWIG 允许向广泛的脚本语言公开 C/C++ 代码，包括Ruby、Perl、Tcl和Python。

不需要其他 OOT （面向对象技术）模块。

1.2 示例

1.2.1 在线（基于SDR）

利用SDR，软件无线电，即Software Defined Radio，SDR。通俗来讲，SDR就是基于通用的硬件平台上用软件来实现各种通信模块。

这将使用连接的HackRF或USRP捕获完整的Iridium频段并将检测到的突发信号解调为帧。如果同时有许多突发，它将使用抽取来跟上。

1.2.2 离线（基于文件）

在离线模式下处理文件，并通过命令行提供所需的选项。

1.3 从原始数据中提取铱星数据帧

可以离线处理文件或将数据流式传输到工具中。提取器可以读取配置文件，它还接受来自命令行的参数。

有些选项只能通过配置文件使用，其他选项只能通过命令行使用。如果不使用配置文件，则可以提供文件名以从文件中读取样本。如果未指定（配置）文件，则从标准输入读取样本。

1.3.1 配置文件

配置文件分为几个部分。每个部分都以“[section-name]”行开头。

1.3.2 `osmosdr-source` 部分

如果此部分存在，则实例化 `osmosdr-source`

|选项名称 |说明 |

|-----------|------------------------------ --------------|

| `sample\_rate` |源应该运行的采样率|

| `center\_freq` |以 Hz 为单位的源中心频率 |

| `gain` | (RF)-以 dB 为单位的增益 |

| `if\_gain` |中频增益 (dB) |

| `bb\_gain` | 基带增益 (dB) |

| `bandwidth` |基带滤波器带宽 (Hz) |  
  
BB gain：基带增益baseband frequency gain  
IF gain：中频增益Intermediate frequency gain  
RF gain：射频增益Radio frequency gain  
一般都是先将基带信号调到中频，再调到高频的载波上，成为射频信号发射出去。  
  
1.3.3 命令行选项

可以使用命令行选项代替配置文件。如果还指定了配置文件，则命令行选项优先。

1.3.3.1 `-o`、`--offline`：离线处理

默认情况下，如果可用的计算能力跟不上，提取器将丢弃样本或突发。如果有一个已经录制的文件，请使用`-o`、`--offline` 选项来不丢弃连拍。在这种情况下，提取器将暂停读取文件（或输入流），直到它可以再次处理更多样本。

1.3.3.2 `-D`、`--decimation`：抽取

此选项可在输入流由基于突发的组件处理之前对其进行抽取和通道化。当许多突发同时出现时，这有助于减少所需的内存带宽。如果您在联机操作期间丢失突发，请使用此选项。

抽取必须是偶数。在内部，多相滤波器组将用于通道化输入频谱。每个通道都将通过所选的抽取进行抽取。为了解决多普勒频移，信道相互重叠。为了提供所需的额外采样率，需要创建一个多于所需的通道并激活过采样。这导致输入带宽 \* (1 + 1/抽取) 的总输出带宽。  
 多普勒频移（Doppler Shift）是多普勒效应在无线电领域的一种体现。其定义为：由于发射机和接收机间的相对运动，接收机接收到的信号频率将与发射机发出的信号频率之间产生一个差值，该差值就是Doppler Shift。  
 不建议使用小于 4 的抽取。抽取输入信号可以提高实时性能，但不建议用于离线处理。

在离线处理期间，它往往成为主要瓶颈。

1.3.3.3 `-c`：中心频率

源或文件的中心频率（以 Hz 为单位）。

1.3.3.4 `-r`：采样率

源或文件的采样率。

1.3.3.5 `-f`：输入文件格式

| File Format | `iridium-extractor` format option |  
|----------------------------------------------------|-----------------------------------|  
| complex uint8 (RTLSDR) | `rtl` |  
| complex int8 (hackrf, rad1o with hackrf-transfer) | `hackrf` |  
| complex int16 (USRP with specrec from gr-analysis) | `sc16` |  
| complex float (GNURadio, `uhd\_rx\_cfile`) | `float` |

RTL-SDR 这是一个低廉的家用消费档次的 DVB-T USB 接口的接收机的创新发明，一个完全的软件定义无线电。  
  
1.3.3.6 `-q`：队列长度

对于每个通道（默认情况下有一个），一个队列中充满了检测器检测到活动的样本。默认情况下，每个队列的长度为 500 帧。可以使用此选项调整队列的长度。

1.3.3.7 `--debug-id`：输出特定突发的调试信息

检测到的每个突发都会分配一个 id。对于解码为帧的突发，它在输出数据中显示为“I:xxxxxxxxxxx”。

设置此选项（例如`--debug-id=23`）以输出突发编号23的调试信息。调试信息包括：

- 在 `stdout` 上调试打印。

- 原始样本文件写入`/tmp/signals`。

1.3.3.8 `--file-info`：文件信息

在输出数据中手动设置文件信息字段（第二个字段）。如果未使用此选项，则将使用当前时间。

1. 解析模块

1 示例

假设提取器的输出已写入“output.bits”，即可以解码。如果想加快这个步骤，可以安装 `pypy` 并运行。

2 帧提取

有关如何从原始数据中提取 Iridium 帧的信息，可以参阅接收模块。

3 语音解码

要收听语音电话，需要一个 AMBE 解码器。

有两种选择：

1. 使用tnt的开源 AMBE 解码器；
2. 从固件二进制文件中提取 AMBE 解码器。

更简单的选择是使用 tnt 的 AMBE 解码器。如果要创建位正确的输出，可以使用提取的解码器。这两个选项之间几乎没有区别。确保 `ir77\_ambe\_decode` 或 `ambe` 在你的 `PATH` 中。确保工具包的主文件夹在你的 `PATH` 变量中。

语音解码步骤：

1. 使用 `iridium-parser` 解码您捕获和解调的位并将结果放入一个文件中；
2. 使用 `stats-voc.py` 查看捕获的语音帧流；
3. 向左和向右单击一次以选择一个区域；
4. `stats-voc.py` 将尝试使用 `play-iridium-ambe` 脚本解码和播放选定的样本。

4 主要成分

4.1 解析器

获取解调位并尝试将它们解析为可读格式，支持一些不同的输出格式（`-o` 选项）。

4.2 mkkml

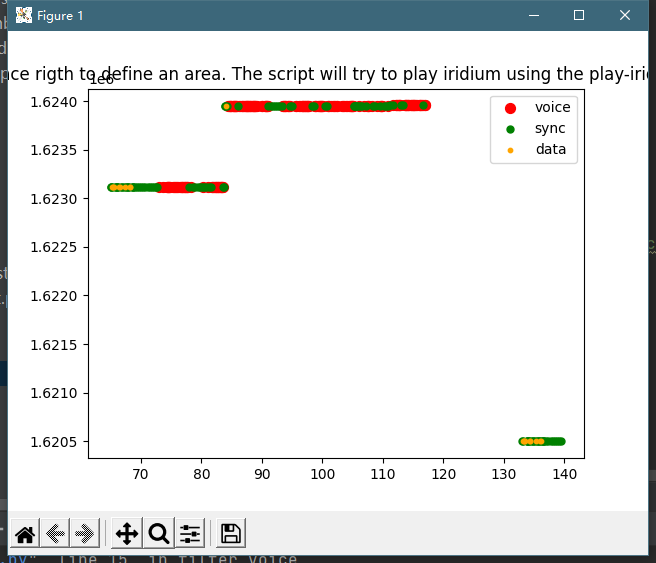
将 IRA 帧转换为可在谷歌地球中查看的 kml 文件。可以显示卫星轨道，或创建卫星位置和下行链路位置的热图。

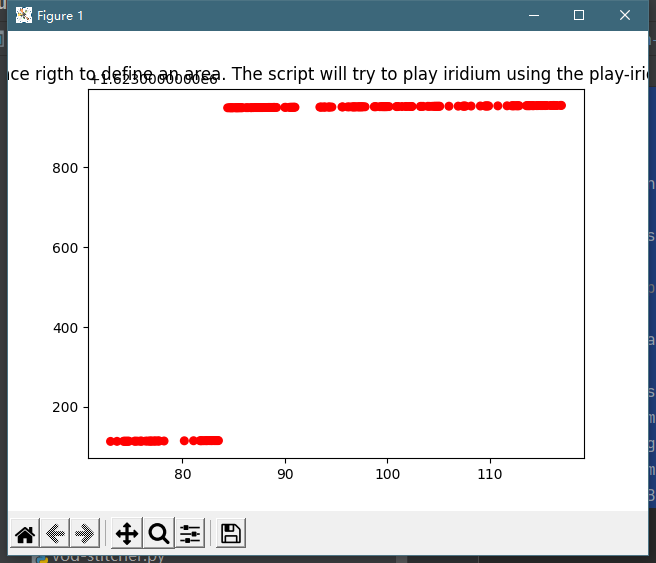
4.3 重组器

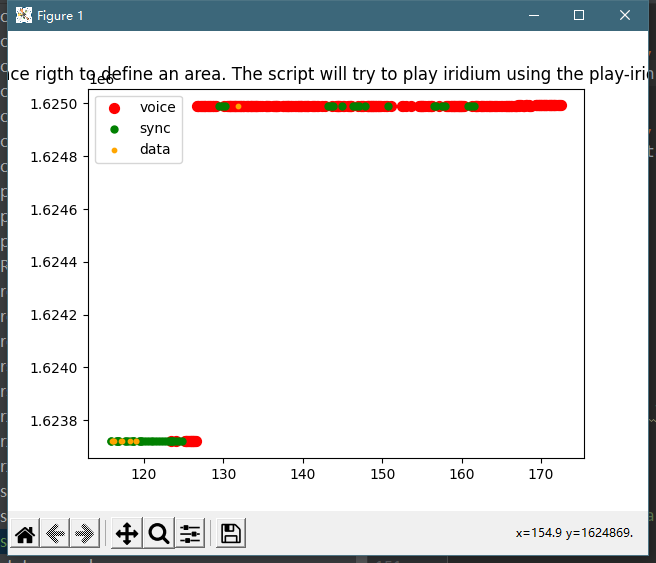
获取解析后的位，并将它们重新组合成更高级别的协议。使用 `-m` 选项支持不同的模式。

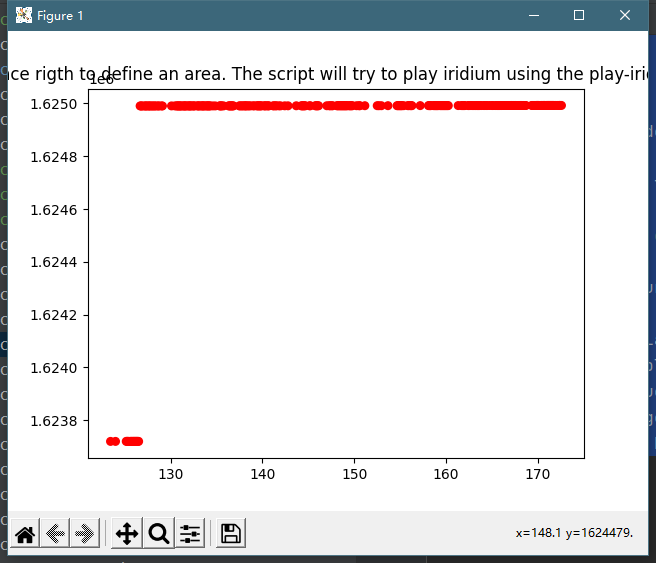
目前支持的模式有：

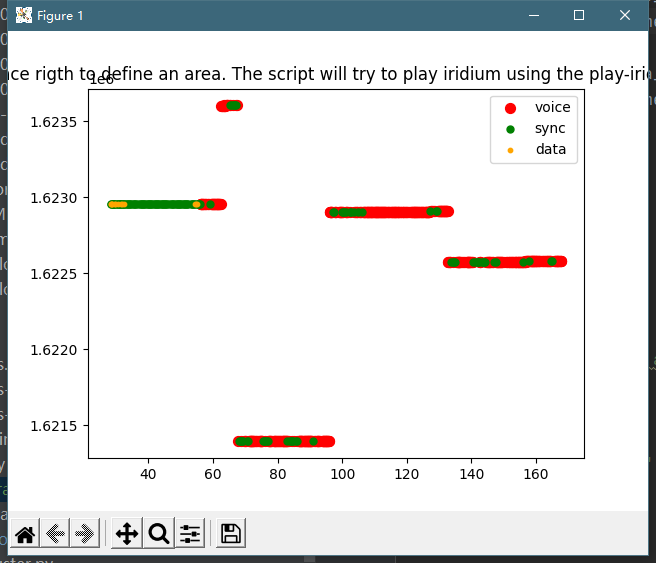
1. `ida` - 以十六进制输出 Um 第 3 层消息
2. `lap` - GSM 兼容的 L3 消息作为 GSMtap 兼容的 `.pcap`
3. `page` - 寻呼请求（环形警报频道）
4. `msg` - 寻呼机消息
5. `sbd` - 短突发数据消息
6. 实验结果

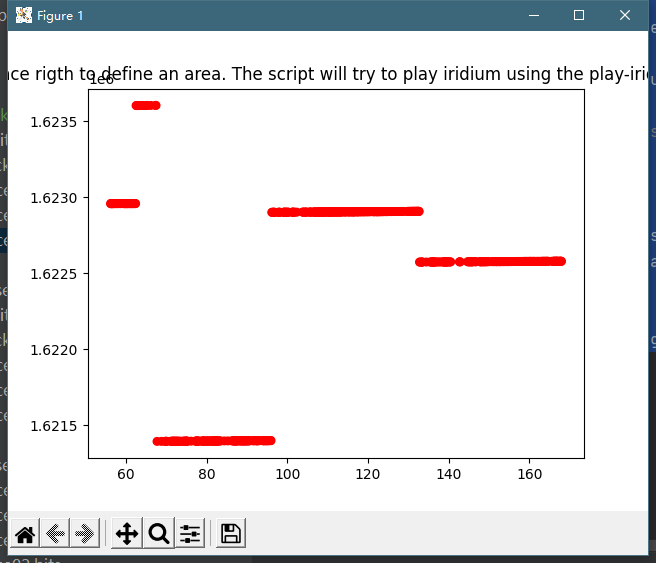












解析语音见附件：

1. output01-voice.wav
2. output02-voice.wav
3. output03-voice01.wav
4. output03-voice02.wav
5. output03-voice03.wav
6. output03-voice04.wav