**卡调系统部署说明**

# 1.矿卡调度系统

使用 Python3 编写的程序，用于使用基于行程价值的启发式算法在露天矿山作业中管理矿卡调度。

# 2.如何安装

## 2.1 环境要求

至少安装Python 3.6

依赖项：sqlalchemy、redis、numpy、datetime、mysql-connector、psycopg2

## 2.2 安装python3.6

（如下介绍是针对于Linux操作系统）

下载并安装python3.6.5版本。

1. 首先安装python依赖的库

Centos:

yum install zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel tk-devel gcc make

ubuntu:

sudo apt-get install zlib1g-dev libbz2-dev libssl-dev libncurses5-dev libsqlite3-dev libreadline-dev tk-dev libgdbm-dev libdb-dev libpcap-dev xz-utils libexpat1-dev liblzma-dev libffi-dev libc6-dev

1. 把Python3.6下载到/usr/local目录

# wget https://www.python.org/ftp/python/3.6.5/Python-3.6.5.tgz

# tar -xzvf Python-3.6.5.tgz -C /tmp

# cd /tmp/Python-3.6.5/

# ./configure --prefix=/usr/local

1. 修改配置，防止出现pip3找不到TLS/SSL的问题出现：



参考链接：

https://blog.csdn.net/Pc620/article/details/86163275

https://blog.csdn.net/hobohero/article/details/54381475?utm\_source=copy%A0

1. 执行make编译命令

# make

# make altinstall

(make altinstall用于防止替换/usr/bin中默认的python软连接)

如果在安装过程中遇到zipimport.ZipImportError: can't decompress data; zlib not available的问题，执行下面两条命令，安装zlib、zlib-devel依赖库即可。

# yum install zlib zlib

# yum install zlib zlib-devel

1. 更改/usr/bin/python链接和pip3链接

# cd /usr/bin

# ln -s /usr/local/bin/python3.6 /usr/bin/python3

# ln -s /usr/local/bin/pip3.6 /usr/bin/pip3

至此，python 3.6安装完毕，但是此时在命令窗口输入python3，即可打开python3的命令窗口界面。

## 2.4 创建虚拟环境（可选）

virtualenv是一个能创建隔绝的独立的Python虚拟环境的工具。它能够建立多个相对独立，互不影响的Python工作环境。也可以直接在python主环境进行配置，请跳过2.4, 直接安装依赖库(2.5)。

1. virtualenv 安装

# pip install virtualenv

1. 创建虚拟环境

安装好virtualenv后，使用virtualenv创建一个虚拟环境：

# virtualenv watodisp # watodisp为虚拟环境的目录名称，名称可以自定义

1. 激活虚拟环境

进入到虚拟环境bin 目录下：

# cd watodisp/bin

执行以下命令启动虚拟环境：

# source activate

启动之后命令行界面会出现一个(watodisp)，这就说明我们启动虚拟环境成功了，可以在当前虚拟环境下安装依赖库。



## 2.5 安装依赖库

**pip3 install** sqlalchemy

pip3 install redis

pip3 install numpy

pip3 install datetime

pip3 install mysql-connector

pip3 install psycopg2

pip3 install pulp

## 2.6 程序运行

在依赖齐全的(虚拟)环境下，进入项目目录下执行：

python3 realtime\_dispatch.py