Dataset for the article:

Scheduling space-to-ground optical communication under cloud cover uncertainty

Mateusz Polnik[1], Ashwin Arulselvan[2], and Annalisa Riccardi[1]

[1]Department of Mechanical and Aerospace Engineering, University of Strathclyde, Glasgow
[2]Strathclyde Business School, University of Strathclyde, Glasgow

1 Introduction

The data set contains a collection of problem instances for scheduling Satellite Quantum Key Distribution (SatQKD). The objective of the problem is to schedule data transfers from a satellite orbiting Earth to a network of ground stations. During a data transfer, the spacecraft sends cryptographic keys which will be then used by the receiver to secure internal communication in the network. The keys should be distributed according to the ground station importance. The optimisation problem is formally stated in [1].

Problem instances were generated using information combined from different sources. In particular, the orbital mechanics and atmospheric models described in [1], official weather forecasts and weather observations published by [2], time series models implemented in [3], and statistic procedures developed by the authors of this manual. Data were preprocessed and organised in a uniform structure designed by the authors.

The dataset is available on the Open Data Commons Open Database License (ODbL) [4].

The document is organised as follows. Section 2 explains the content of the data set. The file format of the problem instances is described in Section 3, which closes the document.

2 Data Set Files

The data set is distributed as a ZIP archive which contains SatQKD problem instances saved in the JSON format.

Each instance starts someday between September 2019 and March 2020, which was the time we downloaded weather forecasts from [2]. The file name indicates the start day, i.e., the file named *week_2019-10-01.json* contains a problem instance whose scheduling horizon commences at the 1st of October 2019. The length of the scheduling horizon for all problem instances in the data set is 120 hours.

3 Problem Instance File Format

Listing 1 presents a high-level structure of a problem instance.

Listing 1: Example format of a problem instance.

```
{
  "metadata": [...],
  "stations": [...],
  "forecasts": {...},
  "var_model": [...],
  "mean_variance_model": {...}
}
```

Overall, data which constitutes a problem instance is split into several conceptual groups. Sections below describe each group in more detail.

metadata

Metadata contains general information about the problem instance and configuration parameters used to create it. Example format of this section is presented in Listing 2.

Listing 2: Example format of the metadata group.

```
"metadata":
[
  ["observation_period", {
    "begin": "2020-Jan-26 12:00:00",
    "end": "2020-Jan-31 09:00:00"
  }],
  ["switch_duration", "00:00:30"],
  ["scenarios_number", 256],
  ["scenario_generator", "past_error_replication"]
]
```

The section consists of the following elements:

observation_period Scheduling horizon the problem instance covers,

switch_duration Time the satellite has to wait after stopping a data transfer to one ground station and before starting sending data to a different ground station,

scenarios_number Number of alternative weather scenarios the problem instance contains,

scenario_generator Algorithm used to generate alternative weather scenarios.

stations

The section contains the list of ground stations and information about their communication windows with the satellite. A communication window is described by its start and end time, the elevation angle between the ground station and the spacecraft at a given time, and the key transfer rate in the cloud-free line of sight (i.e., with no cloud cover). Communication windows, elevation angles, and key transfer rates were computed using models presented in the paper [1]. Furthermore, we assumed the same initial parameters of the satellite.

Listing 3 illustrates an example format of this section.

Listing 3: Example format of the stations group.

```
"stations":
[
  {
    "communication_windows":
    [
      {
        "elevation": [15.005, ..., 15.267],
        "key_rate": [0.059, ..., 0.208],
        "period":{
          "begin": "2020-Jan-26 23:58:16",
          "end": "2020-Jan-27 00:04:59"
        },
      },...
    ],
    "initial_buffer": 64,
    "key_consumption": 0,
    "station": "London",
    "transfer_share": 0.465
  },...
]
```

For each ground station, the section contains a record in the following format.

communication_windows The list of time windows. Each time window contains the following elements:

elevation Vector of elevation angles updated every second of the communication window,

key_rate Vector of key transfer rates updated every second of the communication window,

period Begin and end of the communication window,

initial_buffer Initial number of keys the ground station holds,

key_consumption Amount of keys the ground station consumes during the scheduling horizon,

station City where the ground station is located,

transfer_share Importance of the ground station in the network.

forecasts

The section contains information about cloud cover conditions during the scheduling horizon: the official cloud cover forecast, alternative cloud cover scenarios, and real cloud cover conditions observed during the scheduling horizon.

Listing 4 illustrates an example format of the section.

Listing 4: Example format of the forecasts group.

```
"forecasts":
{
  "forecast":
```

```
  {
    "index": ["2020-Jan-26 12:00:00", ..., "2020-Jan-26 18:00:00"],
    "stations":
    [
      {
        "station": "London",
        "cloud_cover": [71.0, ..., 100.0]
      },...
    ]
  },
  "real": {...},
  "scenario_0": {...},...
}
```

The section is saved in the format of a dictionary, i.e., a map of key-value pairs. The key *forecast* points to the official cloud cover forecast. The real cloud cover conditions are stored under the *real* key. The keys generated according to the format *scenario_number*, where *scenario* is the prefix and *number* is an integer, point to alternative cloud cover forecasts.

A cloud cover forecasts/observations are stored in the following format:

index Vector of date-times which represents points in time of predictions/observations

stations List of records for each ground station,

- **station** City where the ground station is located,
- **cloud_cover** Vector of cloud cover predictions/observations.

Official cloud cover forecasts and actual weather conditions were extracted from the web service [2].

var_model

The section contains the parameters of the Vector Autoregression Model of cloud cover changes for each ground stations. The time series has order one, i.e., the current state is a linear combination of the former state and a white noise process. Listing 5 presents a snippet of the VAR model for a given problem instance.

Listing 5: Example format of the var_model group.

```
"var_model":
[
  [
    "Birmingham",
    {
      "residual": {
        "value": -0.0,
        "stderr": 25.667,
        "level_0": 0.0,
        ...,
        "level_100": 132.405
      },
```

```
      "const": {"value": 0.469, "stderr": 1.228},
      "Birmingham": {"L1": 0.459, "L1.stderr": 0.017},
      ...
    },...
  ]
]
```

Each ground station has its VAR model which can be extracted using the name of the city where the ground station is located. Every model is saved in the following format:

residual Parameters of the white noise process,

- **value** Should be always set to zero if the model is fitted correctly,
- **stderr** Standard deviation of the white noise process,
- **level_p** Value of the residual which comprises p percent of cloud cover changes between time steps. The element is defined for different values of p, i.e., $p \in [0, 5, 10, \ldots, 95, 96, 97, 98, 99, 100]$.

const Constant of the VAR model,

- **value** Value of the constant,
- **stderr** Standard deviation of the estimate,

ground_station_name Parameters of the VAR model for the given ground station. The model contains one record defined for every ground station,

- **L1** Value of the parameter,
- **L1.stderr** Standard deviation of the estimate.

The parameters of VAR models were computed using the software library [3] for actual weather conditions observed in the year 2018.

mean_variance_model

The section contains mean and variance of cloud cover predictions for each ground stations as well as confidence intervals for each estimator. Listing 6 illustrates an example format of the section.

Listing 6: Example format of the mean_variance_model section.

```
"mean_variance_model":
{
  "confidence": 0.95,
  "index": ["2020-01-26 12:00:00", ..., "2020-01-26 18:00:00"],
  "Birmingham": {
    "mean": [83.0, ..., 99.941],
    "mean_lower": [83.0, ..., 97.58],
    "mean_upper": [83.0, ..., 102.29],
    "variance": [0.0, ..., 146.684],
    "variance_lower": [0.0, ..., 88.49],
    "variance_upper": [0.0, ..., 222.282]
  },...
}
```

The section contains the following entries:

confidence Value of the confidence interval,

index Vector of date times corresponding to mean/variance parameters,

city Name of the city where the ground station is located,

- **mean** Vector of mean cloud cover for every point in time,
- **mean_lower** Vector of lower bounds on mean,
- **mean_upper** Vector of upper bounds on mean,
- **variance** Vector of cloud cover variance,
- **variance_lower** Vector of lower bounds on cloud cover variance,
- **variance_upper** Vector of upper bounds on cloud cover variance.

Mean cloud cover comes from official weather forecasts [2]. The variance was computed using cloud cover forecasting errors collected between October 2019 and February 2020. Confidence intervals for each estimator were derived using the bootstrap method [5].

References

[1] M. Polnik, L. Mazzarella, M. Di Carlo, D. K. Oi, A. Riccardi, A. Arulselvan, Scheduling of Space to Ground Quantum Key Distribution, EPJ Quantum Technology (2020).

[2] Open Weather Map. Service homepage [online] (2019). Last accessed 25/1/2019.

[3] W. McKinney, J. Perktold, S. Seabold, Time series analysis in python with statsmodels, in: Proceedings of the 10th Python in Science Conference (SciPy 2011), 2011, pp. 96–102. URL `http://conference.scipy.org/proceedings/scipy2011/pdfs/statsmodels.pdf`

[4] Open Knowledge Foundation. Open data commons open database license [online]. Last accessed 22/10/2019.

[5] T. J. DiCiccio, B. Efron, Bootstrap confidence intervals, Statistical Science 11 (3) (1996) 189–212.