Отчет по заданию 4

Выпуск None

Бородин Григорий, группа 420

нояб. 16, 2017

Содержание:

1 Содержательная постановка задачи

Дана система из N модулей. У каждого модуля есть M вариантов. У каждого варианта 2 характеристики:

- надёжность – вещественное значение из интервала `[0; 1]`
- стоимость (натуральная величина)

Требуется выбрать для каждого модуля 1 вариант так, чтобы надёжность всей системы была максимальной при выполнении заданного ограничения на стоимость. Считать, что все модули соединены последовательно. Параллельно в модуле может быть использован 1 вариант.

Предлогается использовать *метод ветвей и границ*

2 Математическая постановка задачи

2.1 Дано

- Дискретное `N`-мерное пространство `STATES` - Каждая компонента вектора в этом пространстве принимает значения из множества `{0, 1, ..., M - 1}`
- Натуральное число `MAX_COST`
- Отображение `COST` из пространства `STATES x {0, 1, ..., N - 1}` в множество натуральных чисел
- Отображение `REL` из пространства `STATES x {0, 1, ..., N - 1}` в интервал `[0, 1]`

2.2 Задача

Найти такой вектор `BEST_STATE`, что значение функции `STATE_REL` будет максимально при выполнении условия `STATE_COST(BEST_STATE) <= MAX_COST`, где

- отображение `STATE_REL` определяется так:

```
STATE_REL(STATE) = REL(STATE[0]) * REL(STATE[1]) * ... * REL(STATE[N - 1])
```

- отображение `STATE_COST` определяется так:

```
STATE_COST(STATE) = COST(STATE[0]) + COST(STATE[1]) + ... + COST(STATE[N - 1])
```

Если таковых несколько, то ответом является любой.

3 Алгоритм решения задачи

3.1 Основная идея алгоритма

Чтобы решить задачу, необходимо обойти все пространство `STATES`

Отбрасывание подпространств по `COST`

Если для вектора `STATE` выполняется условие:

```
COST(STATE[0]) + COST(STATE[1]) + ... + COST(STATE[K - 1]) > MAX_COST
```

то при решении задачи можно сэкономить несколько операций, и не рассматривать ни один вектор `X`, такой что `STATE[i] == X[i]` для всех `i = 0, 1, ..., K - 1`.

Отбрасывание подпространств по REL

Если для вектора STATE выполняется условие:

```
REL(STATE[0]) * REL(STATE[1]) * ... * REL(STATE[T - 1]) < BEST_REL
```

где BEST_REL - наибольшая уже известная надежность вектора, то при решении задачи можно сэкономить несколько операций, и не рассматривать ни один вектор X, такой что STATE[i] == X[i] для всех i = 0, 1, ..., T - 1.

3.2 Реализация алгоритма

Список state содержит вектор пространтва STATES.

Обходим пространство в лексикографическом порядке:

```
(0, ..., 0, 0, 0),
(0, ..., 0, 0, 1),
..................
(0, ..., 0, 0, M)
(0, ..., 0, 1, 0)
..................
(M, ..., M, M, M)
```

В случае, когда по первой части вектора можно сразу сказать, что он не является решением, пропускаем обход этого подпространства размерности N - K

4 Экспериментальное исследование

4.1 Генерация входных данных

С помощью отдельного скрипта на питоне была сгенерирована 1000 примеров входных данных для каждого значения параметра k.

Параметры запуска

- 6 модулей (N = 6)
- 5 вариантов в каждом модуле (M = 5)
- Цена вариантов - случайное число из {1, 2, 3, 4, 5}
- Порог цены - k * N, где k из {2, 2.2, 2.5, 3, 4}

4.2 Гипотеза

Надежность последовательно соединенных модулей определена как произведение надежностей этих модулей. Оценим среднее значение надежности.

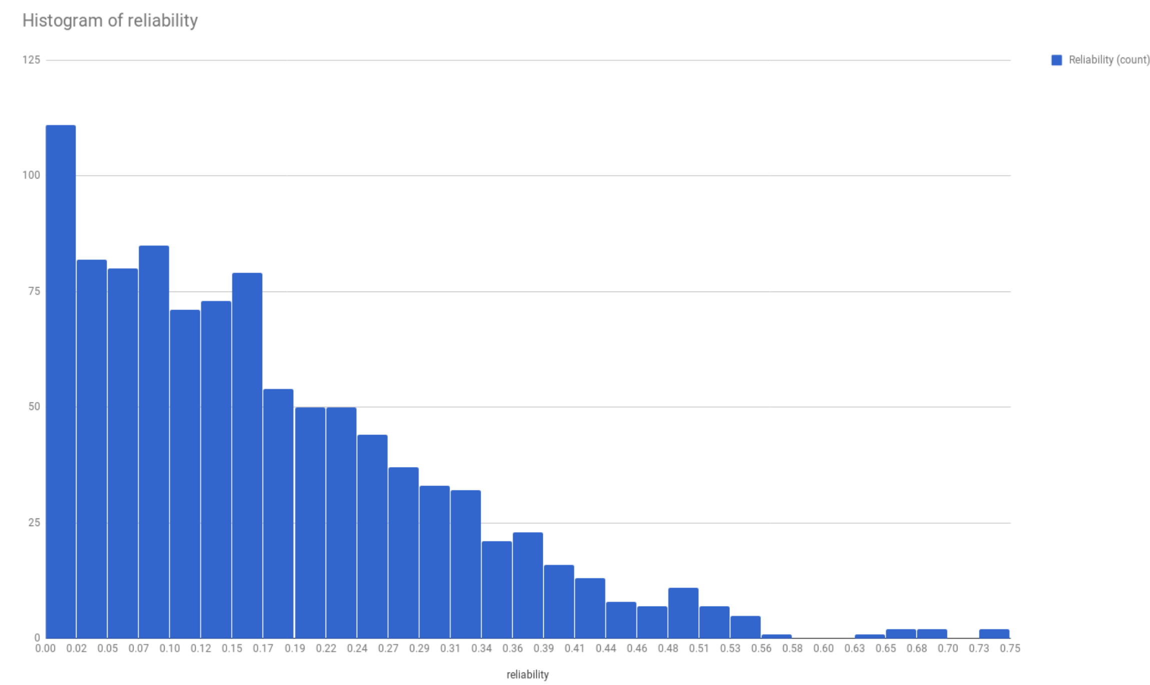
Исходные данные распределены однородно в интервале [0, 1]. Ограничения по цене мы не достигнем, если MAX_COST == 5 * MODULE_COUNT, потому что значение COST для каждого модуля не превышает 5.

Тогда без ограничения по цене мы можем сказать, что основная часть (больше 80%) распределения будет находится в интервале [0, 0.5] с вероятностью больше 99%. То есть вероятность того, что надежность системы будет в интервале [0, 0.5] будет больше 80%.

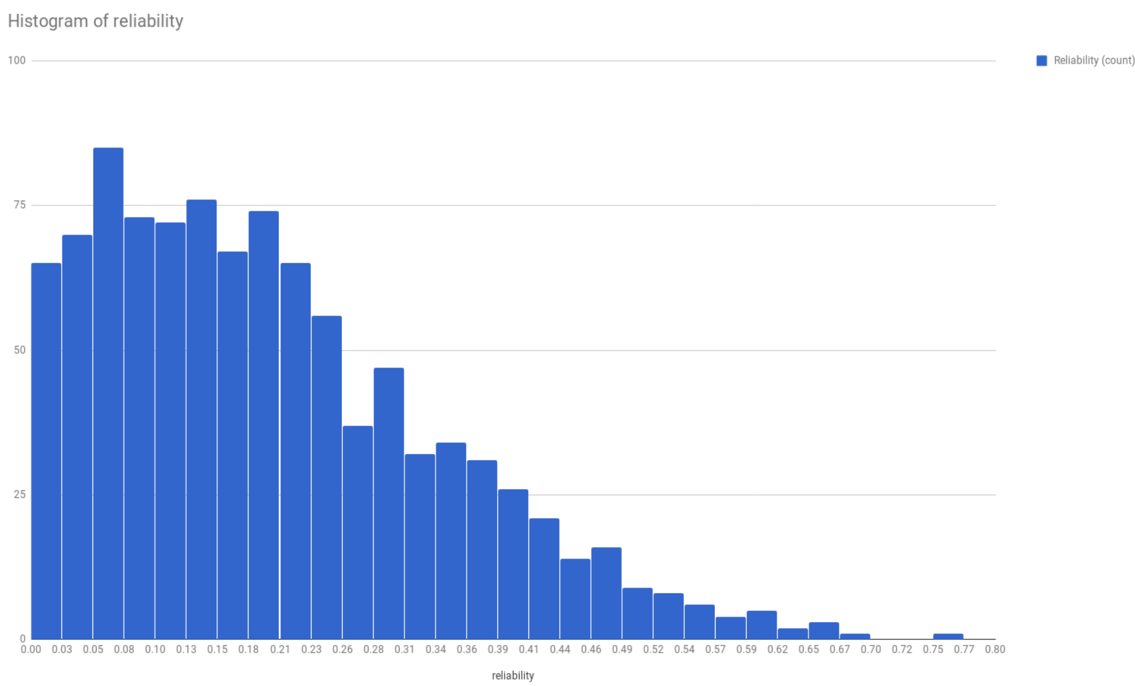
Если же уменьшать MAX_COST, то максимальная надежность может только уменьшиться, так как вводится дополнительное ограничение на стоимость, которое может исключить оптимальный выбор вариантов для обеспечения максимальной надежности.

4.3 Результаты тестирования

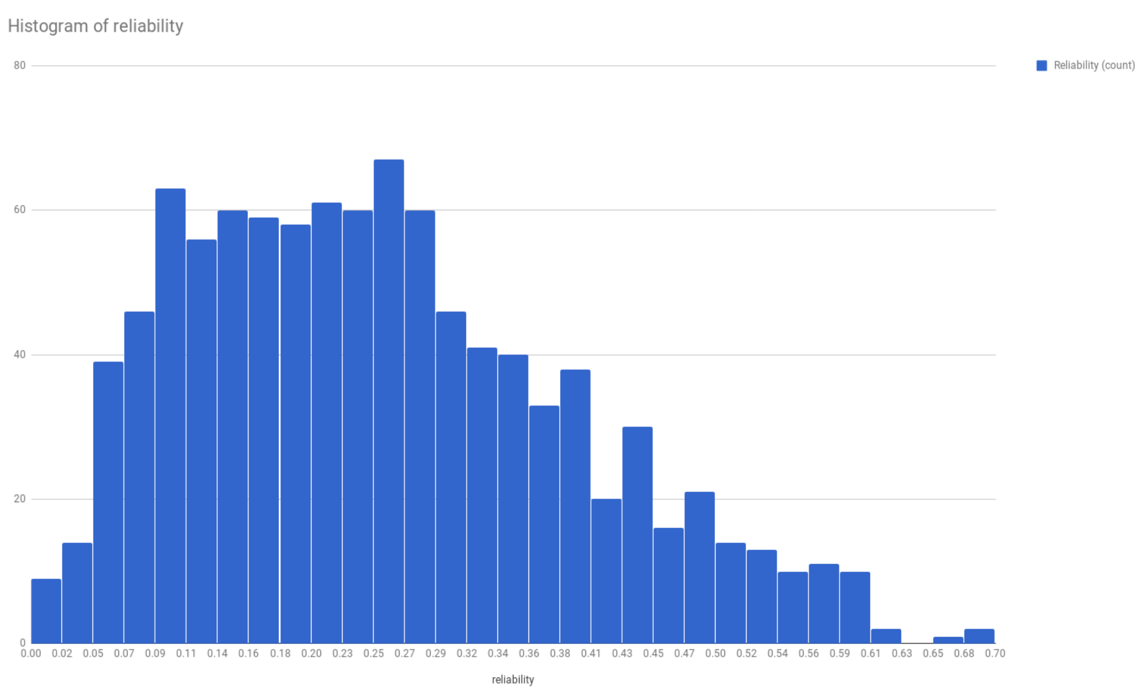
MAX_COST = 2.0 * MODULE_COUNT



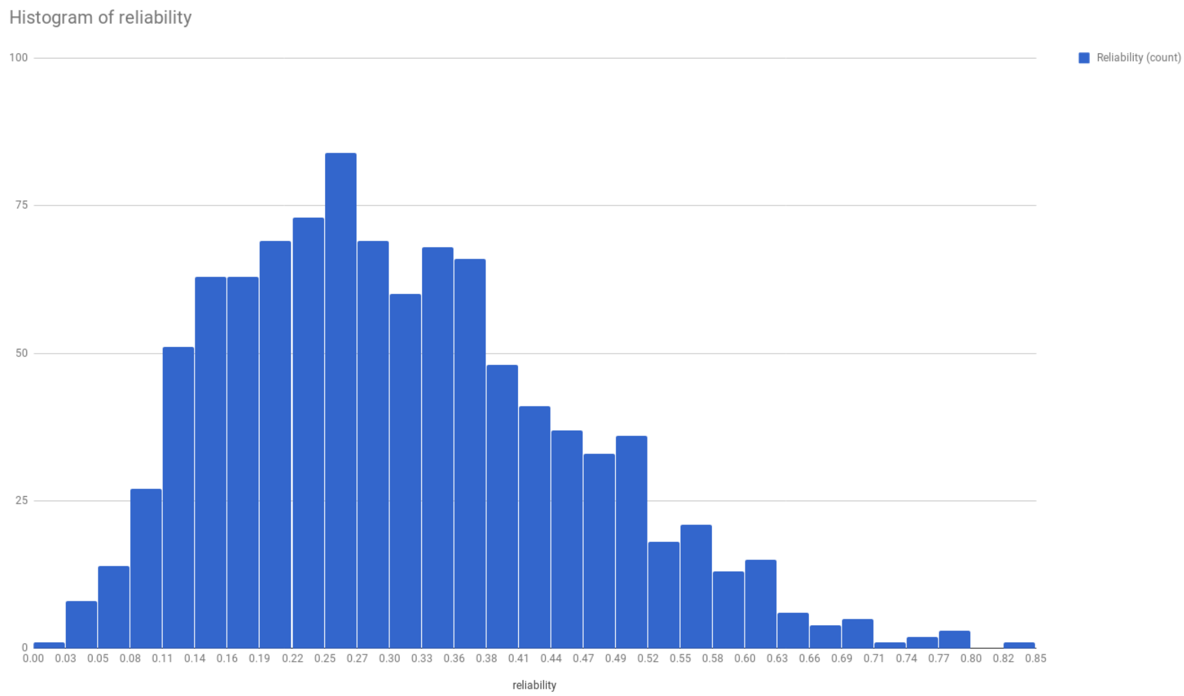
MAX_COST = 2.2 * MODULE_COUNT



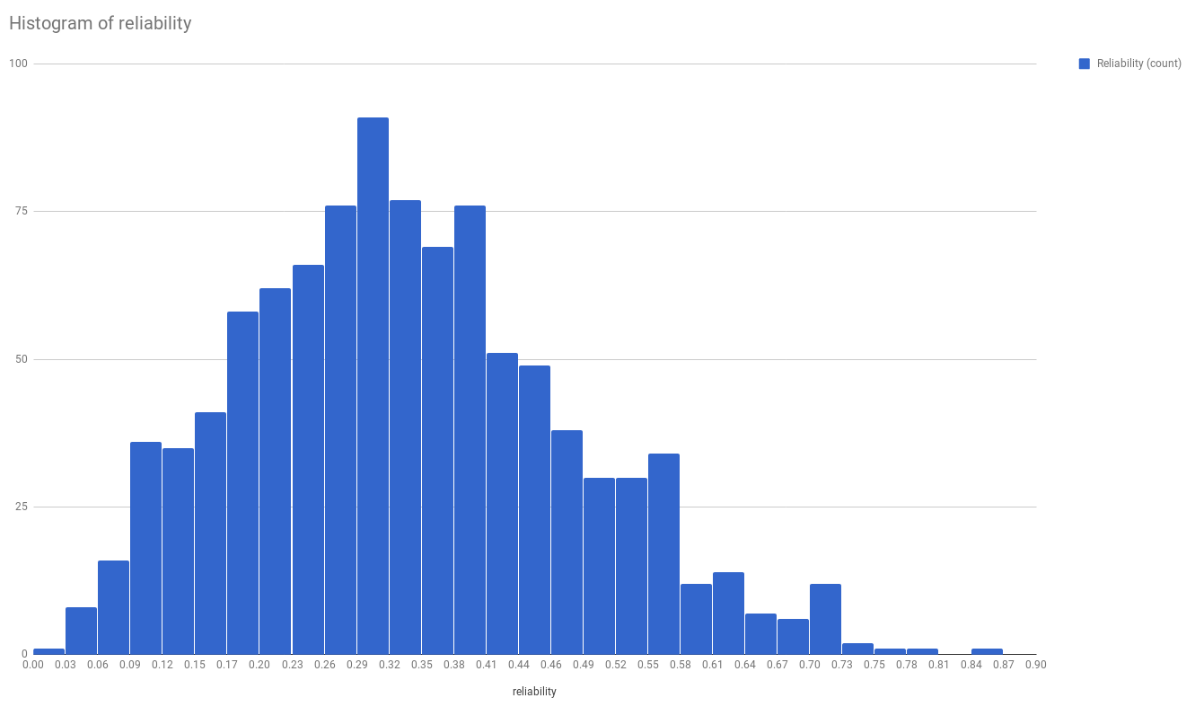
MAX_COST = 2.5 * MODULE_COUNT



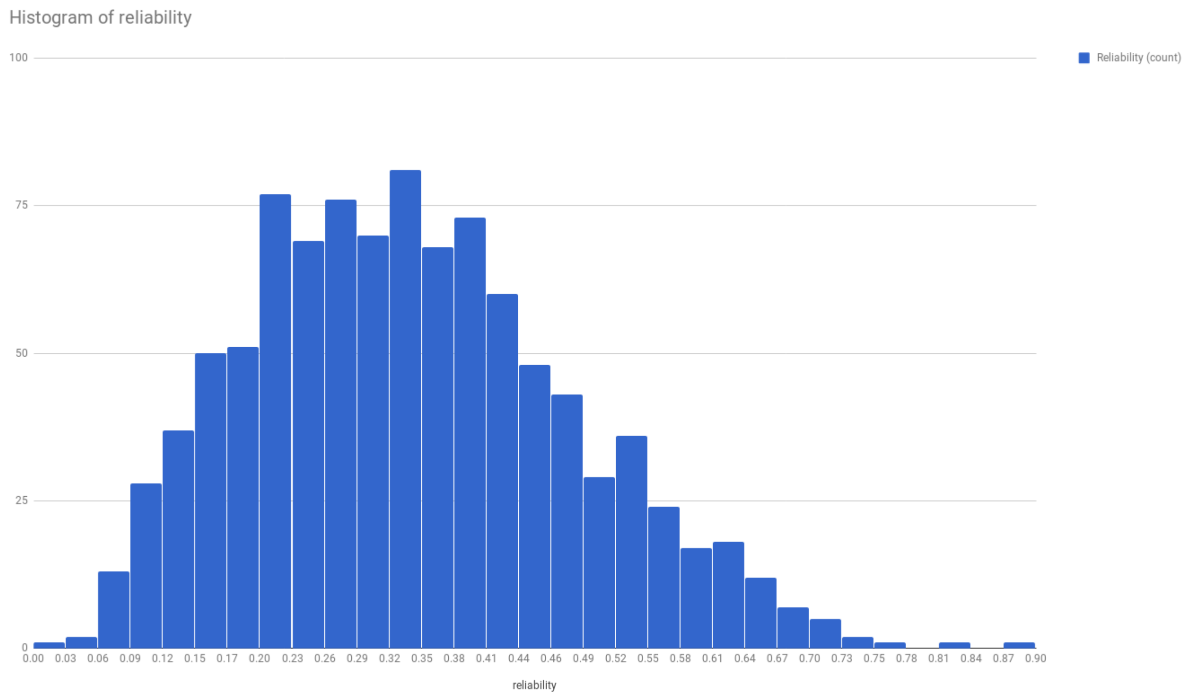
MAX_COST = 3.0 * MODULE_COUNT



MAX_COST = 4.0 * MODULE_COUNT



MAX_COST = 5.0 * MODULE_COUNT



5 Выводы

Гипотеза полностью подтвердилась данными.

По результатам исследования можно даже делать предположения о виде распределения надежности. На графиках видна схожесть с гамма-распределением.

6 Приложение

6.1 Схема представления данных

Данные представлены в формате `XML`. Главный блок - `sample`. Внутри блока `sample` перечисляются блоки `module`, соответствующие модулям. Внутри блока `module` перечисляются элементы `variant`, соответствующие вариантам модуля.

Аттрибут `max_cost` элемента `sample` определяет константу `MAX_COST`.

Аттрибуты `cost` и `reliability` элементов `variant` определяют значение функций `COST` и `REL` соответственно для соответствующего модуля и его варианта.

6.2 Пример входных данных


```xml
<sample max_cost="12">
  <module id="0">
    <variant cost="2" id="0" reliability="0.9644566048434823" />
    <variant cost="5" id="1" reliability="0.8754047459921095" />
  </module>
  <module id="1">
    <variant cost="2" id="0" reliability="0.9340265766458995" />
    <variant cost="4" id="1" reliability="0.5100501134090891" />
  </module>
  <module id="2">
    <variant cost="2" id="0" reliability="0.3513748003407241" />
    <variant cost="3" id="1" reliability="0.40156212172181494" />
  </module>
</sample>
```