motivating_example_notebook

October 7, 2023

1 Motivating Example from Section II

(Presented in "CTA: A Correlation-Tolerant Analysis of the Deadline-Failure Probability of Dependent Tasks" (RTSS '23))

This notebook automates and replicates the computations presented in the motivating example from Section II of our paper, "A Correlation-Tolerant Analysis of the Deadline-Failure Probability of Dependent Tasks," which has been accepted for presentation at the RTSS '23 conference.

We start with necessary imports.

[14]:
```
import numpy as np
from scipy.stats import rv_discrete
from probability_operations.convolution import fast_sum_multiple_rvs
import joint_execution_abstraction.joint_distribution as jd
from itertools import product
```

1.1 Setup

Let us define the two tasks τ_1 and τ_2 and define their possible execution time values in an arbitrary time unit (TU).

- τ_1 can execute for 1, 3, or 5 TUs, while
- τ_2 can execute for 2, or 8 TUs.

[15]:
```
C_1_values = np.array([1, 3, 5]) # possible values for execution-time␣
↪distribution of task 1
C_2_values = np.array([2, 8]) # possible values for execution-time distribution␣
↪of task 2
```

Next, let us define the joint probability distribution of their execution times, as presented in Fig. 1. in the paper.

[16]:
```
                                  # C_2
joint_pmf = [[0.005, 0.01, 0.01],  #  8
             [0.96, 0.005, 0.01]]  #  2
       # C_1   1      3      5
joint_dist = jd.JointRvDiscrete(joint_pmf, C_1_values, C_2_values)
```

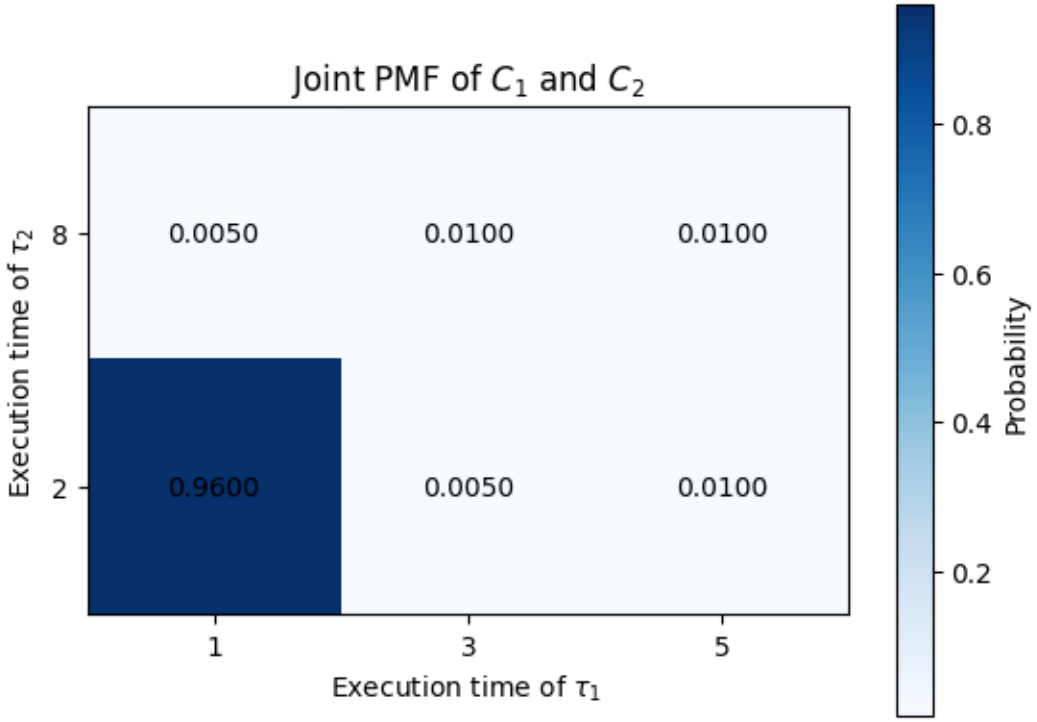
```
joint_dist.pretty_print_joint_pmf() # print the joint pmf
```

```
C_1     0.9650  0.0150  0.0200
------------------------
8       0.0050  0.0100  0.0100  |       0.0250
2       0.9600  0.0050  0.0100  |       0.9750
------------------------
        1       3       5       |       C_2
```

Next, we visualize the joint probability mass function of execution behavior, which is an alternative representation of Fig. 1, and the above printed table.

For example, the following plot illustrates that when a job of τ_1 executes for 5 TUs, the probability that τ_2 executes for 2 TUs is 0.01.

[17]:
```
joint_dist.plot_joint_pmf(x_label='Execution time of $\\tau_1$ ',␣
↪y_label='Execution time of $\\tau_2$', title='Joint PMF of $C_1$ and $C_2$')
```



1.2 Ground-truth worst-case deadline-failure probability (WCDFP)

Let us compute the ground-truth worst-case deadline-failure probability (WCDFP) for τ_2, whose deadline is $D_2 = 10$.

[18]:
```
D_2 = 10 # deadline of task 2
ground_truth_WCDFP = jd.compute_ground_truth_DFP(joint_dist, D_2) # compute the␣
 ↪ground truth DFP
print('Ground-truth WCDFP: ', ground_truth_WCDFP)
```

```
Ground-truth WCDFP:  0.02
```

Now, let us explore three different approaches for calculating an upper bound on WCDFP.

1.3 Assuming Independence When There is None

Consider execution-time distributions of τ_1 and τ_2 which will be characterised with two random variables C_1 and C_2.

[19]:
```
C1 = joint_dist.marginal_X # Random variable representing execution-time␣
 ↪distribution of task 1
C2 = joint_dist.marginal_Y # Random variable representing execution-time␣
 ↪distribution of task 2

print('Printing random variable C_1: ')
jd.print_RV(C1)
print('\nPrinting random variable C_2: ')
jd.print_RV(C2)
```

```
Printing random variable C_1:
Execution time: 1, Probability: 0.965
Execution time: 3, Probability: 0.015
Execution time: 5, Probability: 0.02

Printing random variable C_2:
Execution time: 2, Probability: 0.975
Execution time: 8, Probability: 0.025
```

Now, we can estimate WCDFP, assuming independence between C_1 and C_2.

First, we compute the response-time distribution of τ_2, using convolution.

[20]:
```
response_time_RV = fast_sum_multiple_rvs([C1, C2],[1, 1]) # Random variable␣
 ↪representing response-time distribution of task 2
independence_assuming_WCDFP = jd.
 ↪compute_independence_assuming_DFP(response_time_RV, D_2) # compute the DFP␣
 ↪assuming independence
print('Independence-assuming WCDFP estimate: ',␣
 ↪round(independence_assuming_WCDFP, 6))
```

```
Independence-assuming WCDFP estimate:  0.000875
```

1.4 Safe Over-Approximation with pWCET

In this section, we will compute the WCDFP estimate using a safe method that assumes independence, based on the probabilistic Worst-Case Execution Time distribution (pWCET).

The following function, `derive_minimised_DFP_with_pWCETs`, performs the operations listed below:

- Derives all possible partitions of the sample space for the possible evolutions of a single job for either task, using Definition 3 from [8].
- Filters partitions to include only those that exhibit conditional independence among job costs, in accordance with Definition 4 from [8].
- Computes all possible pWCET distributions for both tasks, using Definitions 5 and 6 from [8].
- Performs DFP estimation for all possible pairs of pWCETs and derives the minimised DFP.

[21]:
```
min_pWCET_WCDFP = jd.derive_minimised_DFP_with_pWCETs(joint_dist, D_2)
print('Minimum possible DFP estimate based on pWCETs: ', round(min_pWCET_WCDFP,␣
↪6))
```

```
Minimum possible DFP estimate based on pWCETs:  0.533333
```

1.5 Embracing Correlation with CTA

Now, we will use the proposed correlation-tolerant analysis (CTA) to compute the WCDFP estimate.

[22]:
```
CTA_WCDFP = jd.derive_CTA_DFP(joint_dist, D_2)
print('CTA-based WCDFP estimate: ', round(CTA_WCDFP, 6))
```

```
CTA-based WCDFP estimate:  0.049818
```

1.6 Summary of the WCDFP results

Let us now present all results in a single table.

[23]:
```
print("Ground-truth WCDFP:  \t     ", ground_truth_WCDFP)
print("Independence-assuming WCDFP: ", independence_assuming_WCDFP)
print("Minimum pWCET-based WCDFP:   ", min_pWCET_WCDFP)
print("CTA-based WCDFP: \t     ", CTA_WCDFP)
```

```
Ground-truth WCDFP:           0.02
Independence-assuming WCDFP:  0.0008749999999999423
Minimum pWCET-based WCDFP:    0.5333333333333332
CTA-based WCDFP:              0.049817990670998305
```

As stated in the paper, we can observe that

- `CTA-based WCDFP` slightly over-approximates `Ground-truth WCDFP`,
- `Independence-assuming WCDFP` under-approximates `Ground-truth WCDFP`,

- `Minimum pWCET-based WCDFP` markedly over-approximates `Ground-truth WCDFP`.

2 Showing the CTA parameters

Lastly, we introduce an extended report to the Correlation-Tolerant Analysis (CTA) for calculating the Worst-Case Deadline First Policy (WCDFP) estimate.

The following command outlines the essential parameters required for computing this bound, specifically, the expectation and standard deviation for each task.

[24]:
```
jd.derive_CTA_DFP(joint_dist, D_2, True) # print the CTA-based DFP with␣
↪detailed parameters
```

```
Expected value of task 1       :   1.110000
Expected value of task 2       :   2.150000
Standard deviation of task 1   :   0.606548
Standard deviation of task 2   :   0.936750
Deadline First Policy (DFP)    :   0.049818
```

[24]:
```
0.049817990670998305
```

3 Generating pWCET distributions

Additionally, if you would like to view the partitions of the sample space and see which pWCET distributions are produced from those partitions, run the following command.

The entire computation is based on the paper "What Really is pWCET? A Rigorous Axiomatic Proposal" which has been accepted for presentation at the RTSS '23 conference.

[25]:
```
jd.derive_pWCETs_all_partitions(1, joint_dist, True) # print all possible␣
↪pWCETs for task 1
```

```
Partition: [[(1, 2), (5, 2), (3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          1     3     5
Probabilities: 0.20  0.40  0.40

Partition: [[(3, 8)], [(1, 2), (5, 2), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.33  0.67

Partition: [[(1, 2)], [(5, 2), (3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.33  0.67
```

```
Partition: [[(3, 8)], [(1, 2)], [(5, 2), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.33  0.67

Partition: [[(1, 2), (5, 2)], [(3, 8), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.33  0.67

Partition: [[(1, 2), (5, 2)], [(3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.60  0.40

Partition: [[(3, 8)], [(1, 2), (5, 2)], [(3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.33  0.67

Partition: [[(5, 2)], [(1, 2), (3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(5, 2)], [(1, 2), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 8), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(5, 2)], [(3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2), (5, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          5
```

```
Probabilities: 1.00

Partition: [[(1, 2), (5, 2), (3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.50  0.50

Partition: [[(3, 8)], [(1, 2), (5, 2), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(5, 2), (3, 2)], [(1, 2), (1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.33  0.67

Partition: [[(3, 8)], [(5, 2), (3, 2)], [(1, 2), (1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2), (3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.33  0.67

Partition: [[(3, 8)], [(1, 2)], [(5, 2), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2), (3, 2)], [(1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8), (3, 2)], [(1, 2), (1, 8)], [(5, 2), (5, 8)]]
```

```
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(3, 2)], [(1, 2), (1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(3, 8), (3, 2)], [(1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(3, 2)], [(3, 8), (1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(3, 2)], [(1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 8), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          3     5
Probabilities: 0.50  0.50

Partition: [[(3, 8)], [(1, 2), (5, 2)], [(3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(5, 2)], [(1, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00
```

```
Partition: [[(5, 2)], [(1, 2), (3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(5, 2)], [(1, 2), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(5, 2)], [(3, 8), (3, 2)], [(1, 2), (1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(5, 2)], [(3, 2)], [(1, 2), (1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(5, 2)], [(3, 2)], [(1, 2), (1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 8), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          5
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(5, 2)], [(3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          5
Probabilities: 1.00
```

[25]:
```
[<scipy.stats._distn_infrastructure.rv_sample at 0x177299b90>,
 <scipy.stats._distn_infrastructure.rv_sample at 0x177276810>,
 <scipy.stats._distn_infrastructure.rv_sample at 0x177292990>,
 <scipy.stats._distn_infrastructure.rv_sample at 0x177277850>,
 <scipy.stats._distn_infrastructure.rv_sample at 0x177229a90>]
```

[26]:
```
jd.derive_pWCETs_all_partitions(2, joint_dist, True) # print all possible␣
↪pWCETs for task 2
```

```
Partition: [[(1, 2), (5, 2), (3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2), (5, 2), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2), (3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(5, 2), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 8), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2), (5, 2)], [(3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(5, 2)], [(1, 2), (3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
```

```
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(5, 2)], [(1, 2), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 8), (3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 2)], [(3, 8), (1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(5, 2)], [(3, 2)], [(1, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2), (5, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2), (5, 2), (3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2), (5, 2), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(5, 2), (3, 2)], [(1, 2), (1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(5, 2), (3, 2)], [(1, 2), (1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00
```

```
Partition: [[(1, 2)], [(5, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2), (3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(5, 2), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2), (3, 2)], [(1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8), (3, 2)], [(1, 2), (1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          2     8
Probabilities: 0.33  0.67

Partition: [[(3, 8)], [(3, 2)], [(1, 2), (1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(3, 8), (3, 2)], [(1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(3, 2)], [(3, 8), (1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(3, 2)], [(1, 8)], [(5, 2), (5, 8)]]
pWCET
Values:          8
```

```
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 8), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2), (5, 2)], [(3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2), (5, 2)], [(3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(5, 2)], [(1, 2), (3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(5, 2)], [(1, 2), (3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(5, 2)], [(1, 2), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(5, 2)], [(3, 8), (3, 2)], [(1, 2), (1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(5, 2)], [(3, 2)], [(1, 2), (1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(5, 2)], [(3, 2)], [(1, 2), (1, 8)], [(5, 8)]]
```

```
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 8), (3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 2)], [(3, 8), (1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(1, 2)], [(5, 2)], [(3, 2)], [(1, 8)], [(3, 8), (5, 8)]]
pWCET
Values:          8
Probabilities: 1.00

Partition: [[(3, 8)], [(1, 2)], [(5, 2)], [(3, 2)], [(1, 8)], [(5, 8)]]
pWCET
Values:          8
Probabilities: 1.00
```

[26]:
```
[<scipy.stats._distn_infrastructure.rv_sample at 0x177276a90>,
 <scipy.stats._distn_infrastructure.rv_sample at 0x17726ad50>]
```