Experiment 1:

The objective of this experiment is to run a comparative performance experiment between Algorithm A2 (Stochastic Local Search Algorithm) and A3 (Evolutionary Algorithm).

- **Experiment Design:**
  - **Parameter Settings:**

    For Algorithm A2, we decided to choose two parameters.

    1) Walk probability threshold for generating initial solution
    2) Objective function (0 = slot / 1 = penalty) to optimize

    For both algorithm A2 and A3, some initial solutions are generated using an initial solution generator which builds up an initial solution space. For algorithm A2, when this initial solution is generated, a probabilistic decision is made between two choices. The initial solution can be generated without following any greedy heuristics, or by following a greedy heuristics. If probability >= Walk probability threshold, the initial solution is going to be generated according to an optimizing heuristics that tries to greedily optimize the objective function (slots / penalty). Otherwise the initial solution is going to be generated randomly without following any greedy optimization technique. So lowering the threshold would increase initial solutions in A2 that were built up using a greedy optimization heuristics.

    We start off with a default threshold of 0.6, and then start lowering this parameter (0.25, 0.1) and collect observations. The values were chosen to represent declining threshold, meaning increasing inclination towards building an initial solution in a greedy approach.

    Based on the objective function input (0 = slot / 1 = penalty), both the algorithms can either target to optimize (lower) the number of necessary slots (parameter value 0), or to optimize (lower) the penalty (parameter value 1). As a result, the objective function input is another parameter to our algorithm.

    For each walk probability threshold (higher to lower), we run our A2 algorithm with different objective function parameters. In total, 6 runs of A2 are performed with the following parameter settings for each testing dataset D:

    Threshold = 0.6, objective function = 0
    Threshold = 0.6, objective function = 1
    Threshold = 0.25, objective function = 0
    Threshold = 0.25, objective function = 1
    Threshold = 0.1, objective function = 0
    Threshold = 0.1, objective function = 1

For Algorithm A3, we decided to choose two parameters.

1) Numbers of generations until which recombination or mutation will be performed.
2) Objective function (0 = slot / 1 = penalty) to optimize

Number of generations is increased (25 to 50, 100, 200) by combining the objective function parameter. Number of generations was chosen as a parameter because in the implementation of A3, the design was such that each generation has objective function values better or same as the previous generation. So increasing or decreasing the height of family tree should impact the performance of the Algorithm. And Objective function input was chosen as a parameter because the algorithm, like A2 is designed to optimize (lower) either the number of necessary slots (parameter value 0), or (lower) the penalty (parameter value 1). As a result, the objective function input is another parameter to our algorithm.

In total, 8 runs of A3 are performed with the following parameter settings for each testing dataset D:

Generation number Threshold = 25, objective function = 0
Generation number Threshold = 25, objective function = 1
Generation number Threshold = 50, objective function = 0
Generation number Threshold = 50, objective function = 1
Generation number Threshold = 100, objective function = 0
Generation number Threshold = 100, objective function = 1
Generation number Threshold = 200, objective function = 0
Generation number Threshold = 200, objective function = 1

- **Performance Metrics:**

From the observations, both the number of slots and the corresponding penalty results can be collected. For Experiment 2, we selected only one of these (the number of slots) to determine the performance metric for the algorithms, and do a comparative analysis. As performance metric, **% to best known solution** was chosen with the following calculation: **% to best known solution = (obtained slot result – best known slot result)/ best known slot result * 100.**

This value generates negative when the algorithm generates a better solution than the best known solution for a given problem instance.

The number of slots was chosen to be the sole factor on determining the performance metric for the algorithms in experiment 2, because combining both numbers of slots and penalty to determine performance required assigning appropriate weights to slots and penalty, which was complicated. So instead of focusing on both # of slots and penalty together for measuring performance, only one criterion is chosen for performance measurement. Number of slots are chosen, because from pilot studies, it was found that the penalty values are spread wider than the #of slots. So using the number of slots was more appropriate.

- **Data Set:**
  52 datasets provided by the instructor were selected as the sample data for doing the comparative performance experiment. These datasets were chosen as they have a large number of exam (min number of exams = 100, max number of exams = 5000), and a large number of students (min number of students = 100, max number of students = 71656)

- **Experimental Setup (Extraneous Variables):**
  Each run of each algorithm was performed on each dataset from the given dataset. Each run lasted for 300 seconds. The experiment was performed on a machine with the following configuration: **CPU:** 16*2.13 GHz **Memory:** 128 GB **OS:** Fedora.

Data Table:

Raw data collected from 6 runs of A2 for each testing data set is given below:

A2:

filename	Best Known slots	WP = 0.6 , OBJ Func = 0	WP = 0.6 , OBJ Func = 1	WP = 0.25 , OBJ Func = 0	WP = 0.25, OBJ Func = 1	WP = 0.1 , OBJ Func = 0	WP = 0.1, OBJ Func = 1
AJ_Sample	36	37	62	37	62	36	62
BC_random	7	8	27	8	27	7	27
brantInstance	36	36	94	36	94	36	94
car-f-92	33	34	42	33	42	33	42
car-s-91	36	36	46	36	46	36	46
denseRoster	28	28	30	30	30	28	30
ear-f-83	24	28	49	27	47	26	47
exam_comp_set1	281	281	281	281	281	281	281
exam_comp_set2	429	429	429	429	429	429	429
exam_comp_set3	584	584	584	584	584	584	584
exam_comp_set4	273	273	273	273	273	273	273
exam_comp_set5	681	681	681	681	681	681	681
exam_comp_set6	137	137	143	137	143	137	143
exam_comp_set7	507	507	507	507	507	507	507
exam_comp_set8	255	255	255	255	255	255	255
hec-s-92	19	20	28	21	30	21	30
instance	10	16	32	19	32	19	32
instance2	14	14	32	14	34	14	34

instance30000	91	93	94	93	94	92	94
kfu-s-93	21	23	34	25	34	24	25
limerick	29	30	66	30	29	30	29
lse-f-91	18	19	39	19	22	19	22
mallan	29	29	75	29	75	29	75
official5000	2525	2529	2577	2529	2577	2526	2577
p1	28	28	51	28	58	28	58
rand-gen1	38	37	71	36	71	36	71
rand-gen2	49	49	71	49	71	49	71
rand-gen3	67	67	85	67	85	67	85
rand-gen4	123	124	190	124	190	124	190
rand-gen5	40	42	62	41	62	40	62
rand-gen6	21	23	62	23	60	22	60
rand-gen7	48	46	67	46	67	46	67
rand-instance1	30	31	98	31	98	31	98
random-instance1	223	223	226	223	226	223	226
random-instance10	376	377	377	377	377	377	377
random-instance2	308	308	413	308	413	308	308
random-instance3	318	318	432	318	432	318	319
random-instance4	339	340	342	340	342	339	342
random-instance5	299	299	398	299	398	299	299
random-instance6	275	276	408	274	408	275	277
random-instance7	300	300	303	299	303	300	303
random-instance8	162	162	219	162	219	162	219
random-instance9	277	278	280	278	280	278	280
randomInstance	37	38	72	38	72	37	72
randomtest	88	88	91	88	91	88	91
self-rand	32	33	62	33	56	33	56
sta-f-83-2	35	35	46	35	46	35	46
test	32	28	31	28	31	28	31
testfile	13	14	14	14	14	14	14
tre-s-92	28	23	46	23	37	23	37
ute-s-92	12	14	42	14	41	12	41
yor-f-83	33	34	34	34	34	33	34

Raw data collected from 8 runs of A3 for each testing data set is given below:

A3:

filename	Best Known slots	Gen= 200 OBJ-Func = 0	Gen= 200 OBJ-Func = 1	Gen= 100 OBJ-Func = 0	Gen= 100 OBJ-Func = 1	Gen = 50 OBJ-Func = 0	Gen= 50 OBJ-Func = 1	Gen= 25 OBJ-Func = 0	Gen= 25 OBJ-Func = 1
AJ_Sample	36	37	47	37	47	37	52	37	51
BC_random	7	7	58	8	72	8	62	8	58
brantInstance	36	36	82	36	75	36	71	36	73
car-f-92	33	34	44	34	42	34	44	34	42
car-s-91	36	36	47	36	46	36	47	36	48
denseRoster	28	28	30	28	30	29	30	29	31
ear-f-83	24	28	46	28	29	28	63	28	48
exam_comp_set1	281	281	281	281	281	281	281	281	281
exam_comp_set2	429	429	429	429	429	429	429	429	429
exam_comp_set3	584	584	584	584	584	584	584	584	584
exam_comp_set4	273	273	273	273	273	273	273	273	273
exam_comp_set5	681	681	681	681	681	681	681	681	681
exam_comp_set6	137	137	137	137	137	137	137	137	137
exam_comp_set7	507	507	507	507	507	507	507	507	507
exam_comp_set8	255	255	255	255	255	255	255	255	255
hec-s-92	19	20	26	20	31	20	27	20	30
instance	10	17	20	16	19	17	20	15	17
instance2	14	14	28	14	34	14	34	14	33
instance30000	91	92	92	92	92	92	93	92	92
kfu-s-93	21	24	23	23	25	24	26	23	26
limerick	29	29	29	29	29	29	29	29	29
lse-f-91	18	20	24	20	22	20	23	20	24
mallan	29	29	58	29	61	29	54	29	48
official5000	2525	2522	2587	2522	2577	2522	2574	2522	2596
p1	28	28	38	28	48	28	43	28	43
rand-gen1	38	36	57	37	57	36	55	36	50
rand-gen2	49	49	63	49	66	49	65	49	67
rand-gen3	67	67	77	67	74	67	76	67	80

rand-gen4	123	124	146	124	157	124	144	124	145
rand-gen5	40	40	50	42	50	40	49	40	57
rand-gen6	21	23	56	23	55	23	61	23	50
rand-gen7	48	46	56	46	54	46	58	46	54
rand-instance1	30	31	31	31	31	31	31	31	31
random-instance1	223	224	224	224	226	224	225	224	225
random-instance10	376	376	377	377	377	376	376	377	377
random-instance2	308	308	308	308	308	308	308	308	308
random-instance3	318	318	319	318	319	319	319	318	319
random-instance4	339	339	339	339	339	339	339	340	339
random-instance5	299	299	300	299	299	299	299	299	299
random-instance6	275	276	277	276	277	277	278	277	277
random-instance7	300	302	301	300	303	302	301	329	303
random-instance8	162	162	216	162	223	162	219	162	218
random-instance9	277	280	282	280	280	279	280	279	279
randomInstance	37	37	52	37	57	37	52	37	52
randomtest	88	88	93	88	89	88	91	88	93
self-rand	32	33	46	33	49	33	42	33	45
sta-f-83-2	35	35	38	35	40	35	44	35	38
test	32	28	31	28	31	28	31	28	30
testfile	13	13	14	13	14	13	15	13	14
tre-s-92	28	23	27	23	28	23	28	23	29
ute-s-92	12	13	42	13	14	13	48	13	49
yor-f-83	33	32	34	32	33	32	34	33	33

Comparing performance of each parameter setting P of A2, we observed that the best results are obtained for parameter settings Threshold = 0.1 , OBJ Func = 0.

Similarly comparing performance of each parameter setting P of A3, we observed that the best results are obtained for parameter settings Generation size = 200, OBJ Func = 0.

filename	Best Known slots	A2 slot results	Performance Metric for A2: %From Optimal	A3 slot results	Performance Metric for A3: % From Optimal
AJ_Sample	36	36	0	37	2.777777778
BC_random	7	7	0	7	0
brantInstance	36	36	0	36	0
car-f-92	33	33	0	34	3.03030303
car-s-91	36	36	0	36	0
denseRoster	28	28	0	28	0
ear-f-83	24	26	8.333333333	28	16.66666667
exam_comp_set1	281	281	0	281	0
exam_comp_set2	429	429	0	429	0
exam_comp_set3	584	584	0	584	0
exam_comp_set4	273	273	0	273	0
exam_comp_set5	681	681	0	681	0
exam_comp_set6	137	137	0	137	0
exam_comp_set7	507	507	0	507	0
exam_comp_set8	255	255	0	255	0
hec-s-92	19	21	10.52631579	20	5.263157895
instance	10	19	90	17	70
instance2	14	14	0	14	0
instance30000	91	92	1.098901099	92	1.098901099
kfu-s-93	21	24	14.28571429	24	14.28571429
limerick	29	30	3.448275862	29	0
lse-f-91	18	19	5.555555556	20	11.11111111
mallan	29	29	0	29	0
official5000	2525	2526	0.03960396	2522	-0.118811881
p1	28	28	0	28	0
rand-gen1	38	36	-5.26315789	36	-5.263157895
rand-gen2	49	49	0	49	0
rand-gen3	67	67	0	67	0
rand-gen4	123	124	0.81300813	124	0.81300813
rand-gen5	40	40	0	40	0
rand-gen6	21	22	4.761904762	23	9.523809524
rand-gen7	48	46	-4.16666667	46	-4.166666667
rand-instance1	30	31	3.333333333	31	3.333333333
random-instance1	223	223	0	224	0.448430493
random-instance10	376	377	0.265957447	376	0
random-instance2	308	308	0	308	0
random-instance3	318	318	0	318	0

random-instance4	339	339	0	339	0
random-instance5	299	299	0	299	0
random-instance6	275	275	0	276	0.363636364
random-instance7	300	300	0	302	0.666666667
random-instance8	162	162	0	162	0
random-instance9	277	278	0.36101083	280	1.083032491
randomInstance	37	37	0	37	0
randomtest	88	88	0	88	0
self-rand	32	33	3.125	33	3.125
sta-f-83-2	35	35	0	35	0
test	32	28	-12.5	28	-12.5
testfile	13	14	7.692307692	13	0
tre-s-92	28	23	-17.8571429	23	-17.85714286
ute-s-92	12	12	0	13	8.333333333
yor-f-83	33	33	0	32	-3.03030303
		A2 Mean:	2.189485667	A3 Mean:	2.095919228
		A2 Median:	0	A3 Median:	0
		A2 Mode:	0	A3 Mode:	0
		A2 S.D:	13.34009074	A3 S.D:	10.96259326
		A2 Skew:	5.902778999	A3 Skew:	4.867333283
		A2 Min:	-17.8571429	A2 Min:	-17.85714286
		A2 Max:	90	A2 Max:	70
		A2 variance:	174.5644882	A3 variance:	118.3473717

Statistical Analysis:

Hypothesis: There is a significant difference between the performance of the two Algorithms A2 and A3.

Null Hypothesis: $\mu_{A2} - \mu_{A3} = 0$; i.e there is no performance difference between the two algorithms.

Alternate Hypothesis: $\mu_{A2} - \mu_{A3} != 0$; i.e there is a performance difference between the two algorithms.

Here, each dataset (individual) in our sample population is measured twice – once using the Algorithm A2, and once using the algorithm A3. Hence, two dependent sample differences are being examined.

For both A2 and A3, the mean, median and mode are not equal. Hence, these distributions are not normal. In such a case, a nonparametric test is ideal, and Wilcoxon signed-rank test is performed using statistical package "R". The following result is obtained:

```
        Wilcoxon signed rank test with continuity correction

data:  A2 and A3
V = 64.5, p-value = 0.5861
```

```
alternative hypothesis: true location shift is not equal to 0
```

As the p-value is 0.5861, and is greater than 0.05 significance level, we cannot reject the null hypothesis. As such, we can say that there is not enough evidence available to suggest that there is no performance difference between Algorithms A2 and A3 at the 95% confidence level.

Experiment 2:

- **Description of problem characteristic:**

In A2 and A3, our objective is to optimize either the number of slots, or the cost. In both cases, the maximum number of exams that can be put into a slot will be dictated by two factors, one exam in that slot, and the amount of conflict that exam has with other unassigned exams. So characteristics describing a measurement of conflicting exams can be informative about a given instance. As a way of quantifying this measurement, I selected a characteristics called Exam Conflict Density (R. Qu, 1995) . Exam Conflict density is defined as the following:

Exam Conflict density = $\sum_{i=1}^{n} total\ number\ of\ conflicting\ exams\ for\ exam\ i$ */ (total number of given exams)^2*

This was selected as a characteristics to compare, because when we want to create the most packed slot schedule, the slots with the exams which have high amount of conflict with other exams tend to have low number of exams possible. How the algorithm handles the conflicting exams can affect the overall structure of the packed structure. So intuitively, there could be a relationship between a measurement of conflicting exams in that schedule, and the algorithm performance.

This value is computable from the program input.

In this experiment, we will see that if the performance of the Algorithm A3 can be predicted or explained using this characteristics.

- **Experiment Design:**
  - **Parameter Settings:**
    For Algorithm A3, the number of generations was set to 200, and we selected objective function 0 to optimize.
  - **Data Set:**
    52 instance datasets provided by the instructor were selected as the sample data for this experiment as per instruction.
  - **Experimental Setup (Extraneous Variables):**
    The experiment was performed on a machine with the following configuration: **CPU:** 8*2.6 GHz **Memory:** 32 GB **OS:** Fedora.
  - **Time:**
    The algorithm A3 was run on each instance one time for 5 minutes, and the objective functions (slots and costs) were recorded for each instance.
  - **Performance Metric:**
    Each of the objective function was converted to a performance metric. The following two performance metrics were used:
    Percent of optimal slot = (Obtained slot result-Best Known slot)/Best known slot
    Percent of optimal cost = (Obtained cost result-Best Known cost)/Best known cost
    Negative values indicate the result of that metric is better than the best known solution .

- **Data Table:**

Raw data collected from observation are given here. The performance metric calculation for each instance is given in the accompanied .csv file with this assignment.

problem name	best known slots	Best known costs	A3 Slots	A3 Costs
AJ_Sample	36	877.76	37	2949.45313
BC_random	7	2.05	7	5777.5
brantInstance	36	255.21	36	6113.3125
car-f-92	33	3180.7	34	123347.375
car-s-91	36	3624.6	36	136722.656
denseRoster	28	3453	28	6726.25
ear-f-83	24	2342	28	49387.4375
exam_comp_set1	281	7030.5	281	26395.8128
exam_comp_set2	429	10791	429	24395.8537
exam_comp_set3	584	14167	584	30502.3344
exam_comp_set4	273	13132	273	10103.6329
exam_comp_set5	681	11830	681	25545.1414
exam_comp_set6	137	21330	137	36319.185
exam_comp_set7	507	14674	507	45511.7432
exam_comp_set8	255	6062.8	255	24037.9033
hec-s-92	19	2461.7	20	40057.25
instance	10	7423.5	17	10835.5
instance2	14	52.61	14	1643
instance30000	91	35954	92	52108.8706
kfu-s-93	21	1112.9	24	99013.125
limerick	29	7019.3	29	63569.6875
lse-f-91	18	451.01	20	46728.5
mallan	29	35.9	29	2876.375
official5000	2525	2.13	2522	6829.04859
p1	28	309.77	28	3315.5625
rand-gen1	38	312.96	36	4406.60938
rand-gen2	49	50.07	49	1295.42969
rand-gen3	67	558.01	67	2277.70972
rand-gen4	123	785.4	124	5699.85946
rand-gen5	40	402.15	40	2629.54688
rand-gen6	21	1200.5	23	13575.5
rand-gen7	48	435.57	46	2503.44922

rand-instance1	30	425.89	31	12692.25
randomInstance	37	1535.8	37	2622.65625
random-instance1	223	198.42	224	10358.0243
random-instance10	376	1130.1	376	6331.81908
random-instance2	308	259.77	308	1704.87197
random-instance3	318	272.31	318	1941.15434
random-instance4	339	369.65	339	2944.44242
random-instance5	299	310.86	299	1720.78769
random-instance6	275	414.87	276	2229.02009
random-instance7	300	982.89	302	4067.00659
random-instance8	162	219.16	162	1345.22475
random-instance9	277	48.37	280	4107.40682
randomtest	88	19329	88	29014.5514
self-rand	32	145.55	33	1729.90625
sta-f-83-2	35	9963.4	35	41592.4063
test	32	7950.7	28	14661.5
testfile	13	870.5	13	3779.5
tre-s-92	28	1316	23	44039.625
ute-s-92	12	252.82	13	60952
yor-f-83	33	6822.4	32	22518.0938

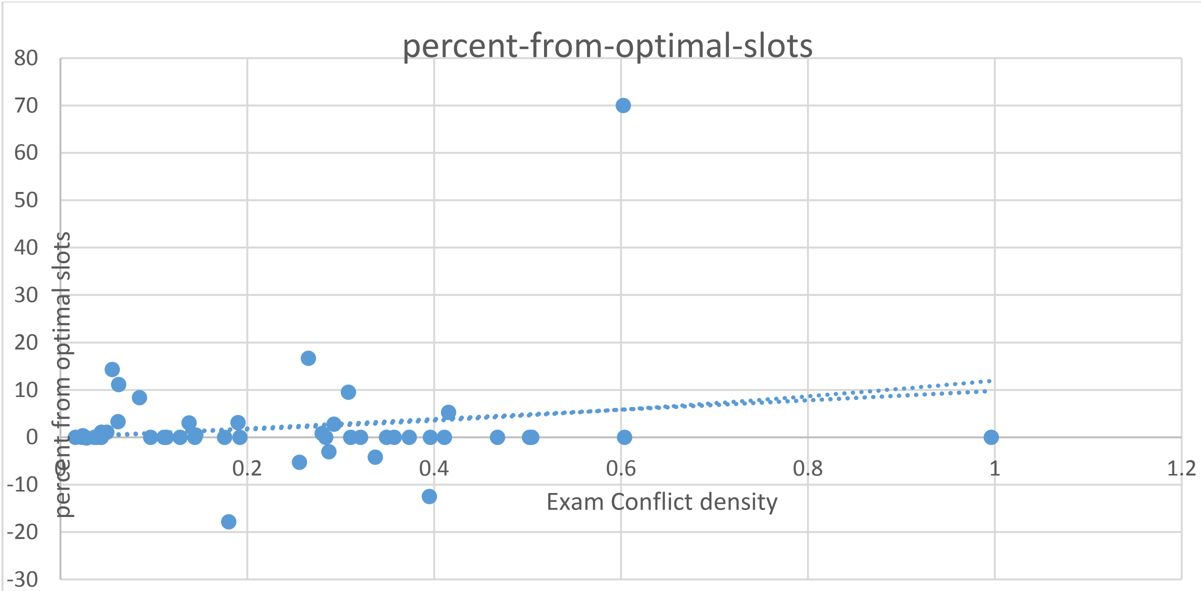
- **<u>Hypothesis:</u>**

  Here, we want to see if there are any relationships between exam conflict density, and the performance of algorithm. So we set up the following hypothesis:

  Null Hypothesis: *There is no relationship between Exam conflict density and performance of the algorithm measured by performance metric percent-from-optimal-slots.*
  Alternate Hypothesis: There is a relationship between *Exam conflict density and performance of the algorithm measured by performance metric percent-from-optimal-slots.*

- **<u>Statistical analysis/Explanation:</u>**

  The data (from the .csv file) can be graphically examined using a scatterplot. Here, the dependent variable (y) is the performance metric (*percent-from-optimal-slots)*, and the independent variable (x) is Exam conflict density. This is so as because we vary the instances to get the performance metric response in this experiment. The following scatterplot is obtained:



The correlation co-efficient r = 0.182245. The correlation coefficient measures the strength and direction of a linear relationship between two variables (here, Exam conflict density and percent from optimal slots). The value indicates that there is a weak positive linear correlation between Exam conflict density and percent of optimal slot, i.e = if the Exam conflict density increases, then performance of the algorithm A3 degrades (positive performance metric indicates degrading performance of algorithm) by a weak relationship.

As we are testing if "any" kind of relationship exists between the two variable, we should go for a two-tailed test. As we are running our algorithm on 52 instances, our degree of freedom is 52-2 = 50.

Here, α = 0.05 for two tailed test and there are n= 50 degrees of freedom. Test statistics t for this value is = 1.310615, which is calculated by the following formula: t = $\frac{r\sqrt{(n-2)}}{\sqrt{(1-r^2)}}$

For two tailed test, when α = 0.05 and there is 50 degrees of freedom, the critical value for t is 2.0086 > 1.310615. So the null hypothesis cannot be rejected.

- **<u>Conclusion:</u>**

The statistical analysis reveals that is not enough evidence available to reject the null hypothesis that there is no relationship between *Exam conflict density and performance of the algorithm measured by performance metric percent-from-optimal-slots.*

Bibliography

R. Qu, E. K. (1995). Benchmark Data Sets in Exam Timetabling., (p. http://www.cs.nott.ac.uk/~rxq/data.htm). Nottingham.