Jack Eadie
jeadie
20865679

Technical Assignment 2

Question 1

Part A

The local search operator used is a local, pairwise swap of cities. Let x = (a,b,c,d,e) be the current solution (whereby the trip from e-> a is assumed). Then the local operator is swapping consecutive cities, in this case producing: Moveset(x) = [(b,a, c,d,e), (a,c, b, d,e), (a,b,d, c, e), (a,b,c,e, d)]. This operation is well defined (preserves a tour) because all elements in original moveset are still included and no additional cities are added or removed. The use of this operator was chosen as a local operator as pairwise permutations of a sequence can construct any transformation from arbitrary tours (because any bijection in the symmetric group of tours can be decomposed from a finite number of 2-cycle permutations; i.e. finite number of swap operations).

Part B

Three different temperature schedules were used for the simulated annealing. The annealing algorithm was tested on tour sizes between 5 and 15 inclusive. For each tour size, 5 problems were tested 3 times. Both the final score and reduction in score were recorded. This testing can be found in timing.py: temperature_schedule_test.

The first temperature schedule was a constant decrease every time a worse state was selected. After a few trials a 10% decrease appeared to be a valid and robust value. The second temperature schedule was a decrease every time a worse state was selected except this decrease was proportional to how many iterations through the annealing was. In practice, the temperature was decreased proportional to (1-t) where t is the decimal proportion of iterations compared to the maximum amount of iterations. This schedule still decreases the probability of selecting a bad move as time progresses but allows for the decreasing constant to increase over time. The last schedule based its temperature decrease on the difference between the current solution cost relative to the maximum cost found in the annealing process. This, similar to above, allows for more exploitation when the current solution is still near a non-optimised starting point. If, for example, the current state space path has high costs (i.e. not decreasing

costs significantly) then the temperature does not decrease significantly (as the current cost is close to its max seen along the path). On the other hand, if a path begins to find lower costs, then the temperature is decreased faster.

The following table and graph summarise the results of the three temperature schedules. 0.9 decrease, t-prop decrease and v-prop decrease denote the three temperature scheduling described above, in order.

Table 1: Average final cost for the three temperature schedules

Number of Cities	**0.9 decrease**	**t-prop decrease**	**v-prop decrease**
5	194.5933554	195.5553307	195.4334542
6	246.8052227	241.2027259	250.5364585
7	265.0043687	260.8843409	281.4904876
8	287.4320209	249.2383107	277.3328449
9	361.6980188	288.5538026	372.3047954
10	416.6304908	331.565644	441.7146357
11	413.5723877	367.3938034	436.1524262
12	442.6434252	406.852256	447.8031587
13	517.7636878	397.7919939	519.1489274
14	550.0614323	456.8354926	554.991296

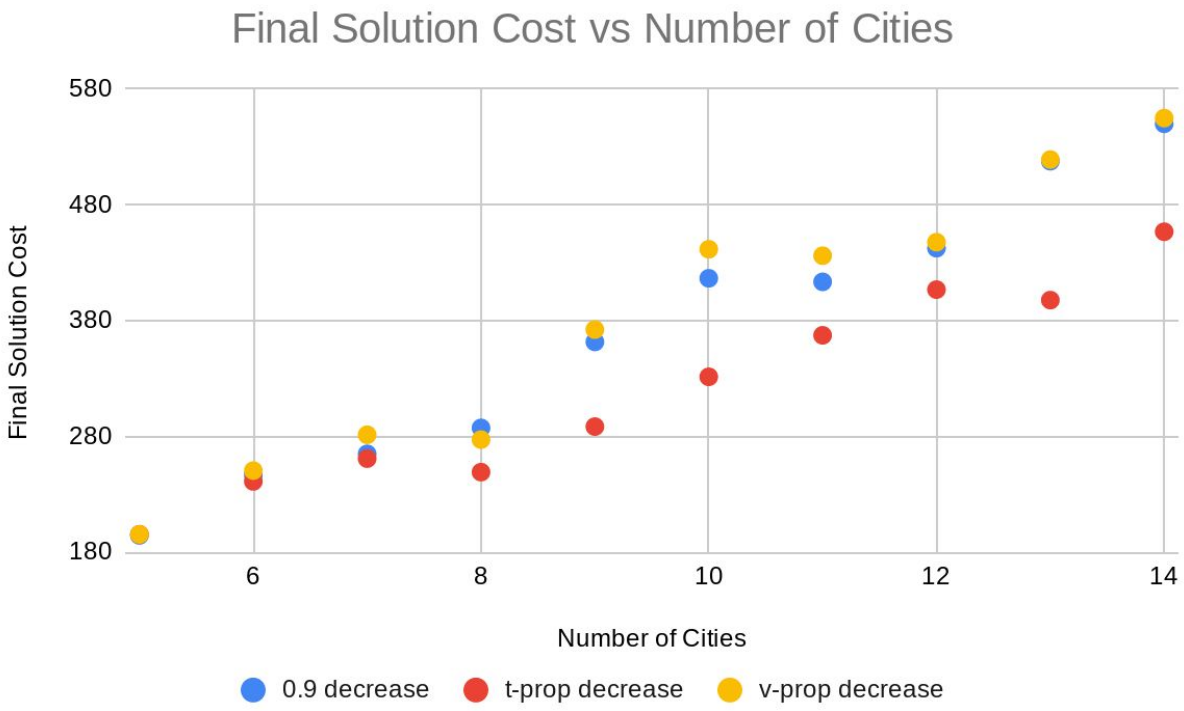


Figure 1: Average Final cost of solutions found via simulated annealing with three different temperature schedules.

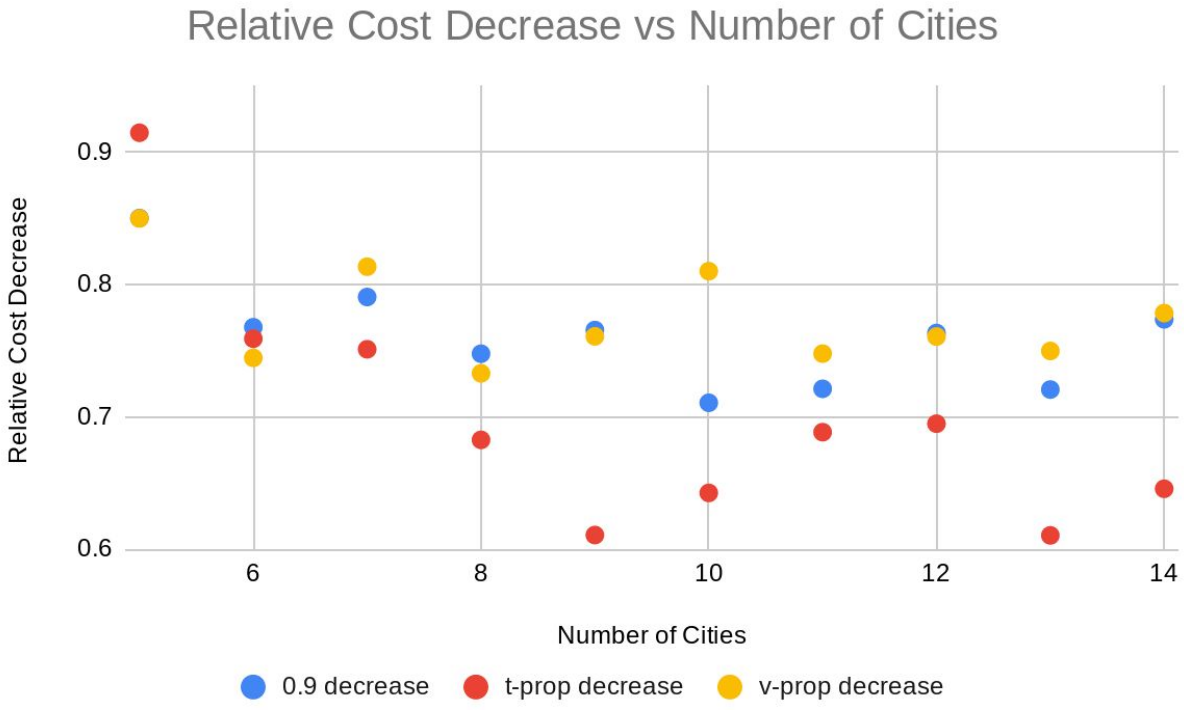
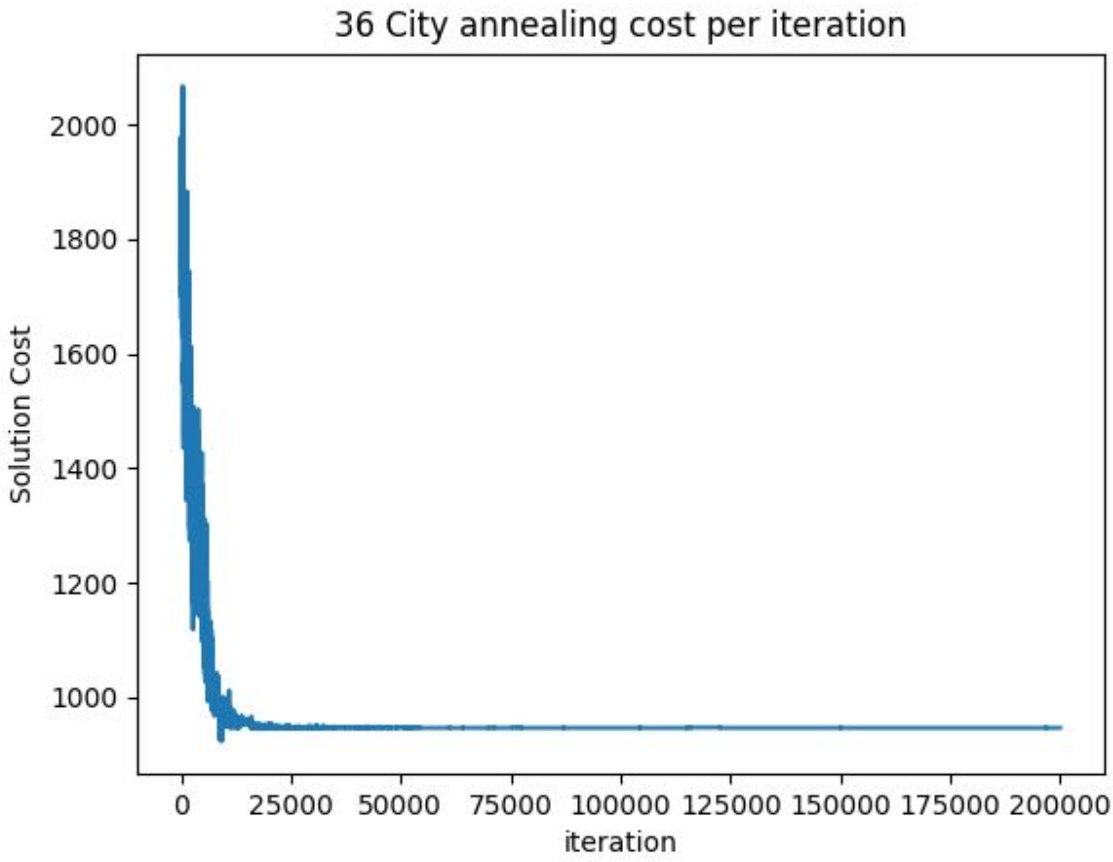


Figure 2: The relative final cost of a solution found via simulated annealing with the three described temperature schedules

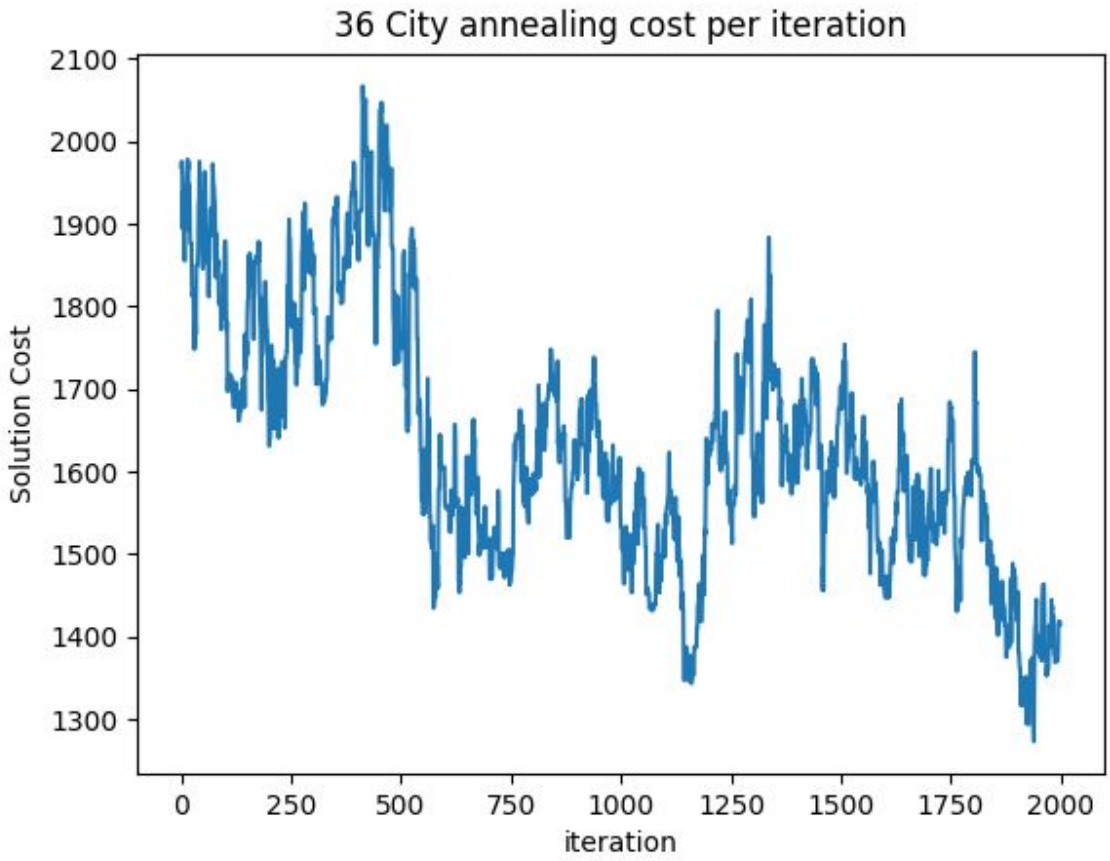
These results suggest that t-prop decrease pose the best temperature schedule to use. However there are some obvious and innate problems with this conclusion based on the preliminary nature of these results. Firstly, no theoretical framework or hypothesis was devised prior to the temperature schedules so these results and reasons for varied success may be identifying the wrong factor. The constant temperature decrease temperature schedule (0.9 decrease) was not strongly validated and the time proportional decrease relied on pre-setting a maximum iterations to base the schedule off. At this stage, the time proportional temperature schedule should be, and is in Part C, used.

Part C

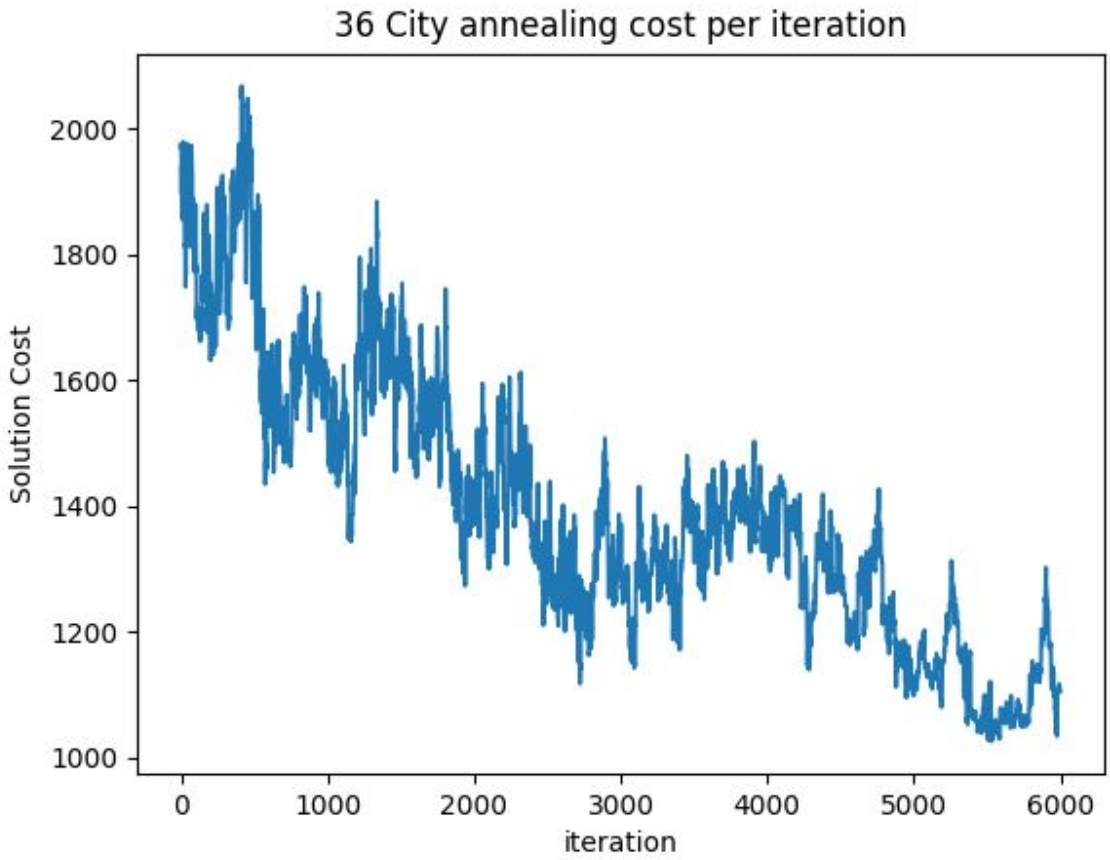
The graph below shows the cost of the solution at each stage of iteration for a full 5 minutes of search.



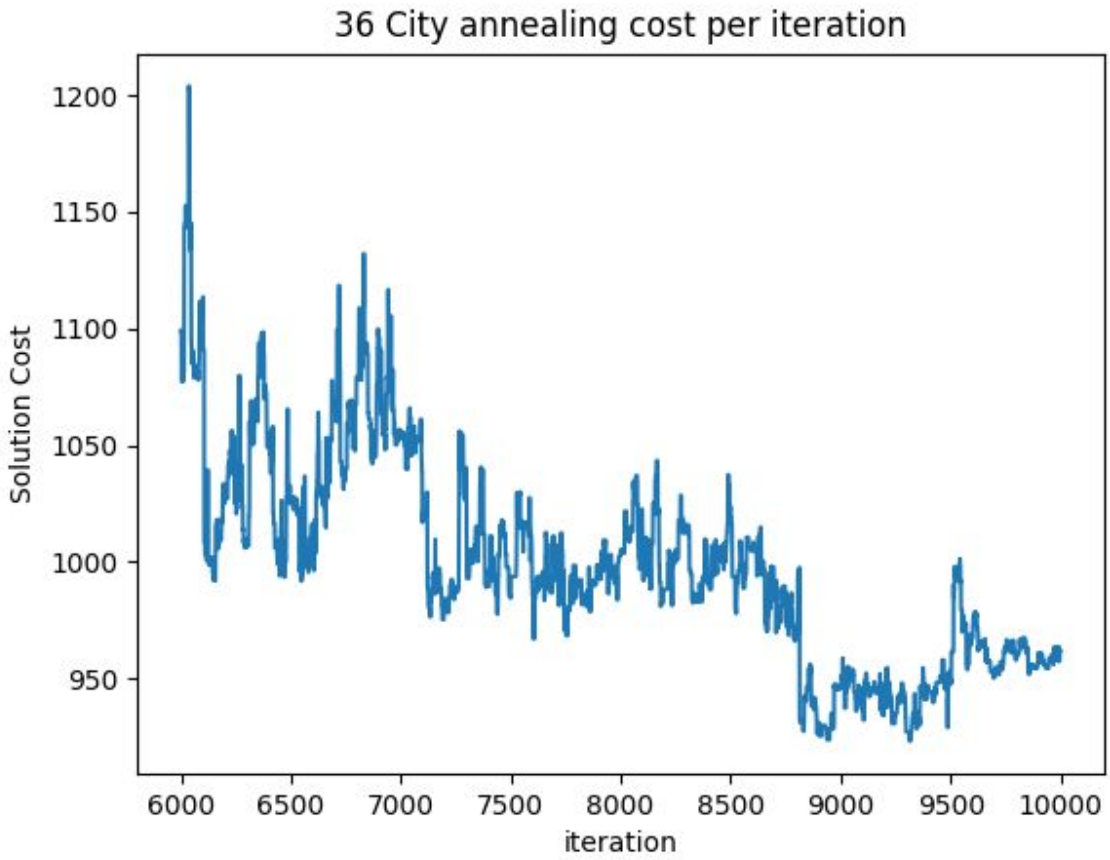
This shows that the majority of annealing was done in a relatively small number of iterations. Looking closer, the first 2000 steps show high exploration of the solution space, the cost jumps significantly in both directions.



However along a larger frame, it is clear that exploitation begins to dominate exploration, tending towards a lower cost solution.



In the expolitation phase, where the probability of making a higher cost move is low, the total variance is, although still present is far lower.



The highest and lowest cost throughout the search were 2067.08 & 922.98 respectively. This is in comparison to the initial and final costs of 1969.39 & 946.33. Clearly simulated annealing's state space traversal moves out of local minima (because it did not finish on the minimum cost found) but also explores locally worse solutions in order to find possibly better solutions (because it clearly went to worse solutions than its inital cost at some point).

Part D

This version of simulated annealing (with the local operator from Part A and any temperature schedule from part b) is complete. This is because local operators begin with a solution to problem (however unoptimal) and any successor function results in another tour (i.e. another solution to the problem).

Part E

Simulated annealing is theoretically guaranteed to find an optimal solution. Although this has assumptions on how many states are traversed. In this implementation after a set number of iteratons, the current tour is returned. Thus, there is no guarantee of optimality.

Question 2

The better_evaluate method was reduced in size so as to prioritise deep traversals of the search tree. Obviously, the method first checks for end of game scenarios and assigns them accordingly. The method then takes the longest chain for each player and gives non-linearly increasing values for chains of greater length. Additionally, for chains that have open slots on either side (and additionally for both) get even further reward. This method is to balance low computational cost whilst mimicking how a user may behave: priorities long chains with open slots and vice versa for the opponent in a non linear way (3 digit chains are not just 50% more valuable than 2 digit chains).

Question 3

The decision tree utilised a training dataset of 132 examples and a testing dataset of 13. On the training dataset, the decision tree had 100% accuracy. This is to be expected in the implementation of the Decision Tree as no early termination or regularisation was used. This means all examples in the training dataset are classified correctly at a leaf node (if they weren't, further branches would be constructed).

On the testing dataset, the above decision tree had an accuracy of 69% (9 of 13 examples). This is above random (50% on binary classification) and at the very least could be used as a weak learner.

Attached (for size reasons) is a full diagram of the constructed Decision Tree.

K >= 5.3
Aniongap >= 9.6
K >= 6.0
K >= 4.2
Aniongap >= 10.6
HEALTHY
COLIC
COLIC
HEALTHY
K >= 5.0
PLA2 >= 153.64301
COLIC
HEALTHY
K >= 4.1
PLA2 >= 167.07899
COLIC
HEALTHY
Na >= 141.0
PLA2 >= 170.78101
K >= 4.0
K >= 3.9
K >= 4.5
Fibrinogen >= 1.95393
COLIC
HEALTHY
COLIC
HEALTHY
COLIC
HEALTHY
K >= 4.1
SDH >= 0.7
COLIC
HEALTHY
K >= 4.0
PCV >= 34.0
K >= 3.9
K >= 4.4
Na >= 137.0
PCV >= 36.0
COLIC
HEALTHY
COLIC
HEALTHY
K >= 3.9
HCO3 >= 27.2
CL >= 100.0
PCV >= 37.0
COLIC
HEALTHY
K >= 4.2
Na >= 141.0
HCO3 >= 28.0
K >= 4.6
K >= 4.2
PLA2 >= 197.60699
COLIC
HEALTHY
K >= 4.7
K >= 3.8
K >= 4.5
Aniongap >= 13.6
COLIC
HEALTHY
COLIC
HEALTHY
Na >= 139.0
K >= 4.7
K >= 3.7
Na >= 140.0
COLIC
HEALTHY
COLIC
HEALTHY
K >= 4.0
COLIC
HEALTHY
COLIC
HEALTHY
K >= 5.2
K >= 4.1
CL >= 102.0
SDH >= 2.0
COLIC
HEALTHY
COLIC
HEALTHY
HEALTHY
K >= 3.7
Na >= 137.0
PLA2 >= 210.72301
COLIC
HEALTHY
K >= 4.4
K >= 3.7
K >= 4.5
PLA2 >= 224.11301
COLIC
HEALTHY
K >= 3.6
CL >= 103.0
K >= 4.5
K >= 4.5
SDH >= 2.1
COLIC
HEALTHY
HCO3 >= 27.4
K >= 4.5
K >= 4.1
CL >= 107.0
COLIC
HEALTHY
K >= 4.6
K >= 4.7
COLIC
HEALTHY
Pulse rate >= 44.0
K >= 3.7
COLIC
HEALTHY
COLIC
HEALTHY
K >= 4.2
CL >= 105.0
K >= 4.2
COLIC
HEALTHY
K >= 3.9
K >= 4.5
PLA2 >= 243.718
K >= 3.5
K >= 4.2
COLIC
HEALTHY
COLIC
HEALTHY
Na >= 143.0
HCO3 >= 29.4
CL >= 104.0
COLIC
K >= 3.6
HEALTHY
COLIC
HEALTHY
HEALTHY
COLIC
K >= 4.5
K >= 3.3
K >= 4.4
K >= 4.3
COLIC
HEALTHY
COLIC
K >= 4.2
FibPerDim >= 20.7584
Na >= 139.0
COLIC
K >= 4.6
COLIC
HEALTHY
K >= 4.2
K >= 4.2
K >= 4.6
K >= 4.6
COLIC
HEALTHY
COLIC
HEALTHY
Endotoxin >= 6.29
K >= 4.3
COLIC
HEALTHY
K >= 4.0
K >= 2.6
COLIC
HEALTHY
COLIC
K >= 2.7
COLIC
K >= 2.8
COLIC
K >= 2.9
Na >= 140.0
K >= 3.0
COLIC
COLIC
Na >= 142.0
K >= 3.1
COLIC
COLIC
Na >= 142.0
K >= 3.2
COLIC
COLIC
HCO3 >= 26.1
K >= 3.3
COLIC
COLIC
Na >= 139.0
K >= 3.4
COLIC
Na >= 142.0
Na >= 141.0
K >= 3.5
HCO3 >= 30.7
COLIC
COLIC
COLIC
Na >= 142.0
COLIC
K >= 3.6
Na >= 144.0
Na >= 140.0
K >= 3.7
COLIC
Na >= 146.0
COLIC
Na >= 141.0
Na >= 135.0
K >= 3.8
COLIC
COLIC
Na >= 142.0
Na >= 138.0
COLIC
K >= 4.0
COLIC
Na >= 145.0
Na >= 140.0
Na >= 140.0
K >= 4.1
COLIC
COLIC
Na >= 141.0
Na >= 143.0
Na >= 139.0
COLIC
COLIC
COLIC
Na >= 144.0
COLIC
COLIC
COLIC
Na >= 140.0
COLIC
COLIC