NAT-DL: Self-study questions & answers for Revision Set 8 (week 10)

Below you can find example questions from previous exams. It unlikely, but not impossible that one of the questions below re-occurs in this year's exam. You may like to check past NAT papers already before this final tutorial, so that you can also discuss other potential exam questions.

Please consider a few hints for answering exam questions. They may not apply to any exams of any other courses, but they will likely apply to this year's NAT exam.

- Preparation is key. There are two ways to prepare:
  1. Answer a choice of questions from past papers, tutorials, and quizzes
  2. Formulate your own questions based on the course material: A typical question is "How does *X* relate to *Y*?", where *X* and *Y* can be principles, facts, algorithms, applications etc. If your question appears very easy, then you can be pleased to know. If it's off moderate difficulty, you may like to search for an answer in the course materials and beyond. If your question is too hard/to detailled/too open-ended, then it is unlikely to occur in the exam, but you may like to keep it in mind for future research
- Past papers contain exam questions of courses from previous years. The course does not remain the same over time, so sometimes you will see a question related to content that was not even mentioned in this year. If you have made sure that it really wasn't mentioned, then you can rest assured that such a question will not occur in this year's exam.
- During the exam, read each question carefully, and plan your answer. A question can have several parts, and in order to get a maximum number of marks, all parts need to be mention in the answer. It is possible to answer all parts in one paragraph and let the marker sort your text into their marking categories, but it is more safe to answer each part in a separate paragraph, even if there may be a bit redundancy.
- If several marks can be achieved for a single question, then you may need to write down several points/examples/reasons/aspects/properties or whatever the question is asking for. Typically (but not always), you get one mark per item, so you can guess how many you need to list.
- If more than one mark is possible for a more challenging answer, then usually one mark is for the plain answer, one is for a good explanation that this is a good answer, one is for limitations of scope of the answer, one is for generalisation, one is for discussion, … but no worries, if this was expected then carefully read the question which needs to indicate this (see above!).

From past exams (in the actual exam, you may like to add more detail in your solutions)

1. Explain the difference between "roulette wheel" selection and a "tournament" selection in genetic algorithms. For what conditions would you prefer on of these selection mechanisms?

   Answer: Roulette wheel is the natural choice if the fitness is explicitly known, tournament is better if only relative fitness is know, e.g. for game-playing agents. Tournament selection tends to give more diversity. It is easier to implement, in particular in parallel implementations of genetic algorithms, and it can be easily adjusted w.r.t. to selection pressure, namely by specifying a probability with which

the winner of a tournament is selected.

2. The Boolean satisfiability problem is as follows: given a formula in conjunctive normal form, find an assignment of Boolean values to the variables in the formula which make the entire formula true. A formula in conjunctive normal form is a conjunction (and) of a set of clauses, with each clause being a disjunction (or) of atomic variables or negated atomic variables. For example:
   (L1 OR L2 OR NOT L3) AND (L3 OR NOT L4 OR NOT L5) AND (NOT L2 OR L4 OR L5)
   (i). Describe how you could apply the canonical genetic algorithm to the Boolean satisfiability problem.
   (ii). How could you improve upon your canonical GA solution for this problem?

   Answer: Since it's a binary problem the approach is straight forward, and you may want to go through the algorithm. Improvements are possible by preprocessing (removing redundant clauses, DPLL algorithm) or by local search (max-ascent hill climbing or random hill climbing). We have also considered neural-network-based variants of the such algorithms, although this may not be required for the answer.

3. Your company has the task of designing controllers for spaceships in the “Killer Asteroid” computer game: asteroids travel across the screen destroying any spaceship they hit, unless the spaceship fires at the asteroid and destroys it first. The spaceships’ aim is to survive for as long as possible. Spaceships may move by firing two thrusters, one on each side of the spaceship; the spaceships are working in a gravity-free environment so firing the thrusters causes a spaceship to translate or rotate or both. A spaceship’s missiles are ejected from its nose, so the spaceship must be pointing in the right direction if a missile is to intersect the path of the asteroid. Assume a spaceship has sensors looking in 8 quadrants around it that can measure quantities like distance to the nearest asteroid in each quadrant, speed of asteroid, bearing of asteroid with respect to the spaceship, etc. Describe how you would use GP to come up with a controller for a spaceship, giving details of and justifying all assumptions you make.
   Discuss any problems your solution might have in evolving good controllers quickly. How would your solution fare if the game is made more complicated, e.g. by requiring the spaceships to avoid each other or allowing them to hide behind planets.

   Answer: We need data from many good players as fitness cases (the algorithm tries to perform as the average over all these players). Given the sensor data in a given time interval, the program has to command the next actions. I don't know the game too well, but GP should deal with it automatically, but I think from here it is standard GP.

4. Give an example of a deceptive fitness function. How can the deceptiveness of a fitness function be measured if the global optimum is known? What is a fully deceptive fitness function? Explain whether the schema theorem applies to the case of deceptive fitness functions.

   Answer: This question can be easily answered after reading the Melanie Mitchell’s book, but it should be possible also without this. The example can be given by a formula: $f(x)=x$ for $x>0$ and $f(0)=100$ for a maximisation problem on [0,99] or by a description e.g. a TSP where the salesperson can take a helicopter which happens to be available at a city which is discouraged by the local heuristics. If the optimum is known, then the deceptiveness can be described as the average local

improvement of the fitness when getting closer to the optimum (which would be negative for in the deceptive case). A fitness function is fully deceptive, if all schemata that sample the optimum are worse (on average over all sampled states) than schemata that don't. The schema theorem applies independently on the deceptiveness, but it is not too helpful to know that a misleading "good" schema is taking over in the population.

5. Formulate the travelling salesperson problem as a combinatorial optimisation.

   Answer: The questions should have had the final word "problem", i.e. refer to COP which we mentioned in Unit 5, slide 43.

6. The Bigbooks book company want to deliver its books to stores around the country. It hires lorries from Toptrucks and delivers several parcels to each store. Each lorry starts at Bigbooks's depot, where it is filled with parcels. Bigbooks wishes to hire as few lorries as possible so the space inside them needs to be used efficiently. The company wishes to use as little fuel as possible, so the distance travelled by the lorries should be as short as possible. You may assume that weight is not an issue, i.e. a fully loaded lorry is not too heavy. How would you use ACO to produce routes for the lorries? In your answer be sure to describe the use you make of heuristics. How can local heuristics be employed in ant colony optimisation problems?

   Answer: This is standard TSP, but with more constraints (in-time arrival etc.). You need to go through the steps of ACO.

7. Describe one crossover and one mutation mechanism that would be suit-able for use with real-valued chromosomes. In each case use an example to illustrate your answer.

   Answer: Crossover should respect the numbers as a whole, mutation rate should be higher for less significant digits of the numbers. Example: Function minimisation. But better than this is local search.

8. State the schema theorem briefly, using either words or an equation. Why might the schema theorem not give an accurate prediction of a GA's performance?

   Answer: Because it gives only a lower bound, because of randomness of the selection (especially for small populations), because of uneven sampling.

9. What characteristics of an optimisation problem do you need to consider when deciding whether to solve it using a GA or GP? What are the advantages and disadvantages of using GP especially when compared to using a GA?

   Answer: Is it a pure search problem or do the "individuals" perform computations. Is the search space discrete or continuous? Is the length of the solution fixed by the problem?
   Advantages/disadvantages: start with checking whether the algorithms fit the above criteria.
   GP: more flexible (if combined with local search for numerical parameters), richer behaviour, practically more successful, ...
   GA more likely to converge to a good suboptimal solution, theoretically better understood, ...

10. How can a genetic algorithm be used to evolve a plan? Give details of all the steps involved. What changes must be made in a genetic algorithm to obtain a steady-state genetic algorithm?

    Answer: There was not much about this in the lecture. A good idea would be to run a GP (see the universal problem in lect 8), but here we are asked for a GA. You will need to encode a sequence of actions. The rest is standard. Consider a robot in a maze where a good solution lead the robot out the maze (some example should be have been given in the exam question)
    Steady state means simply that only one individual is changed at each generation, according to similar methods as in standard GA. Changes are therefore “differential” which is convenient for theoretical methods.

11. How is the diversity of the population maintained in the Differential Evolution (DE) algorithm?

    Answer: Diversity is a problem in DE. Although we have an understanding of the critical parameters, this is valid only on average (i.e. not necessarily representative for a small number of particles) and for a trivial fitness function (i.e. parameters can still get stuck in a local optimum, where the changes are rejected). You would not need to now the formula for the critical parameter values, but mentioning this is useful. The literature present also other methods such as adding noise, adding random particles, or searching for particularly large differences in the population.

12. Particle swarms optimisation (PSO) has been shown to be applicable to similar problems as GA. How does the representation of TSP by PSO relate to GA? Can PSO be expected to outperform GA for the TSP?

    Answer: We have mentioned discrete PSO, but did not study this in detail. The operations needed in a PSO (taking a difference between to vectors, multiplying by a (random) number can be redefined in terms of operations on permutations. A swap of two elements would be the smallest step, such that each solution can be considered as a series of swaps. A difference can then be considered of performing first the swaps of one solution and then the reversed series of swaps from another solution. The multiplication with a number is the execution of a fraction of the swaps (for numbers smaller than 1) or of several times all swaps plus an additional fraction (for numbers larger than 1). It is also possible here to just describe a normal PSO with some discretisation procedure (either in the end or in each step). The questions whether a PSO can be better than a GA is not very clear, because a reasonable PSO can outperform a bad GA, but one would expect that a coarsely defined PSO is less efficient than a GA. The genuinely discrete PSO, mentioned above, however, has been observed to perform quite well and to be well adapted to the problem structure, which is not necessarily true for a GA.

13. Ant colony optimisation (ACO) can be applied to solving time tabling problems. For example, in a railroad company wants to schedule trains such that a maximal number of passengers are transported, delays are minimised, and customer satisfaction is maximised.

    a. Discuss the application of one of the ACO algorithms to this problem. For this purpose describe the problem representation, define a desirability heuristic,

name relevant constraints, formulate a rule for pheromone update, and write down the probability rule.

Answer: Ignoring the particularities of train scheduling, you may simply assume that the question is about moving trains from city to city similar to TSP, however with a fitness function that also includes the other criteria, e.g. persons*kilometers – c_1 * {total number of delayed arrivals} (this is actually often used, although it would make sense to use the mean square deviation of arrival times) + c_2 * {customer satisfaction} (e.g. by direct feedback to staff). You would need to know whether there is more of a chance on some part-routes to accumulate delays, and it is perhaps also a good idea to include the travel time into the fitness. The probability rule and the pheromone update rule (either as a formula or in words) are not really different from the standard form, but it might be good to explain them a bit.

b. Write down the main steps of ACO in pseudocode.

Answer: See ACO lecture. If there are a lot of marks for such a question, you should add some explanation of each step, discuss differences between the variants of the algorithm, however, for an open-book exam questions like this are unlikely to occur.

c. In Ant Colony Optimisation the pheromone trace is reduced in each step by an amount proportional to the evaporation rate. What is the effect of a high or a low evaporation rate on the population and the quality of solutions? Explain how these effects can be used in order to achieve a good performance of the algorithm.

Answer: This is actually a difficult question, as for small numbers of ants the effect of the evaporation is ambiguous. But for an acceptable answer it is sufficient to write that fast evaporation leads to more exploration, slow evaporation to more exploitation. The quality of the solutions depends on the problem: For easy problems not much exploration is needed, to get good solutions, while for difficult problems it may be a good idea to start with fast evaporation and to make is slower over time.

d. The algorithm converges to a good solution only after a very long time. What may have led to the slow convergence? How can the convergence speed be improved?

Answer: This is problematic, because increasing the convergence speed will reduce exploration and can thus lead to worse results, but if you must, you can make the fitness steeper, reduce noise, use a better local search algorithms, or increase the parameter β. See also next question.

e. The algorithm does not converge to a good solution even after very long time. How can the performance be improved?

Answer: A simple option would be to restart, but it might be worthwhile to analyse what is the problem, i.e. checking whether all ants are following the same path (i.e. restart with more noise) or whether the ants are moving completely randomly, i.e. have not discovered any good part solutions (i.e. restart with less noise). It could also help to decrease pheromone evaporation such that more information can be accumulated or to redefine the fitness function to be more (for random ants) or less (for a degenerate colony) steep. It may also be worth checking whether the

exponent of the visibility is not too high, which can lead to an overemphasis of the local aspect which can trap the solutions in a local optimum. You can also change the number of ants. Finally, also this is difficult without referring to a particular example, you can change the representation of the problem or using a different variant of ACO. Suggesting the use of a completely different algorithm is in principle possible, but is not asked for here.

f. Discuss for the present problem how local search is used to improve solutions in ACO.

Answer: Distinguish between the visibility (η) that enters the probability rule, the taboo list, and any additional local search methods. Local search would be the latter. The idea is that large noise (e.g. via τ_{min}) usually counteracts exploitation, but if local search is used before the pheromone is laid, better solution can be found in a greedy manner, such that the pheromone trail is updated by more reliable information. Of course, local search is also directly producing some improvement. You can add here also the examples for local search in TSP or in bin packing.

g. ACO was inspired by the behaviour of real ants. In what way does the algorithm deviate from the biological original?

Answer: This question should not be understood as a question about real ants, although it is fine to refer to some biological facts. More relevant are the following points that reflect properties of the ACO algorithm:

- The separation into part-solutions is an assumption that works well for combinatorial optimisation problems, but not for continuous problems (i.e. real ants live in a continuous world).
- The assumption in some of the algorithms (and these are ones that can be proven to work) that only one ant adds pheromone to the trail is not realistic and appears also to be quite inefficient.
- The fact that pheromone is placed only after the goal is reached is reasonable in optimisation, but does not apply to locally operating agents (or ants).
- There is no interaction among the ants except via stigmergy (i.e. changes in the environment which the ant implement by the pheromone trails)
- The probability rule is great for diversification (if the distribution is not degenerate), but real ants would probably go for the maximum. Consider also the correspondence with reinforcement learning, where the probability rule can be compared to Boltzmann exploration, which is usually not a good choice.