Question 1 (also in Question 1, Theo Lecuyer.pdf):

Algorithm	Time Complexity	Space Complexity	Complete?	Optimal?
BFS	Exponential: O(b^d+1)	Exponential: O(b^d+1)	Yes	Yes
UCS	Exponential: O(b^d+1)	Exponential: O(b^d+1)	It depends, if in finite space yes.	Yes
DFS	Exponential: O(b^n)	Linear: O(bn)	It depends, if in finite space yes.	No
DLS	Exponential: O(b^n)	Linear: O(bn)	It depends, if in finite space yes.	No
IDS	Exponential: O(b^n)	Linear: O(bn)	Yes	Yes
A*	Depends on the heuristic: Exponential in worst case linear in best	Linear: O(bn)	Yes	Yes

Question 5:

a) What were the engineering advances that led to Deep Blue's success? Which of them can be transferred to other problems, and which are specific to chess?

In a sense, Deep Blue revolutionized how we look at and solve different AI and machine learning problems today. One of the biggest things I noticed, early in the paper, was the emphasis on the search algorithms and how they searched. Instead of using a traditional uniform search algorithm, they utilized a non-uniform search that would not only search much deeper but prioritize which moves may be more tactical or critical while not searching as deep in others. This could be applied to other problems where certain areas involve high-dimensional spaces where it is not effective to search the whole area. The paper also talked about an evaluation function that recognized 8000 different patterns. This function is one that, at least theoretically, is likely similarly used in the realm of autonomous vehicles. This is a very impressive function that accounts for the fact that parts of the environment are out of the agent's control. Another thing talked about in the paper is the single-chip chess search engine. This is another piece of deep blue that led to rapid search and evaluation of positions. This piece of deep blue is mostly specific to chess and would be difficult to transfer to other applications.

b) AlphaZero is compared to a number of modern game-playing programs, such as StockFish, which work similarly to Deep Blue. The paper shows that AlphaZero is able to defeat StockFish even when it is given only 1/100 of the computing time. Why is that? Please frame your answer in terms of search and the number of nodes evaluated.

The first thing that came to mind in terms of AlphaZero and its ability to compute very quickly was its deep neural network. Since the neural network is highly trained based on different starting positions it has a lot of options of moves it can make. AlphaZero has a sophisticated evaluation function that allows it to in a sense estimate the value of board positions and their associated probabilities of winning. So even when AlphaZero explores way fewer moves per second, it has much stronger moves for all of those evaluations. Going along with more on the neural network, AlphaZero uses Monte Carlo Tree Search instead of other traditional algorithms. The biggest thing here is the change in the number of nodes evaluated. Using the neural network AlphaZero can play out random games from the current position to help guide where it has to search. This results in the program spending less time searching, searching fewer nodes, and leading to the best moves without needing to calculate every single possible outcome.