Fakultät für Informatik
Facoltà di Scienze e Tecnologie informatiche
Faculty of Computer Science

Advanced Algorithms

Written Examination

2nd July 2014

FIRST NAME		LAST NAME	
STUDENT NUMBER		SIGNATURE	

Instructions for students

Write First Name, Last Name, Student Number and Signature where indicated. If not, the examination cannot be marked.

Use a pen, not a pencil.

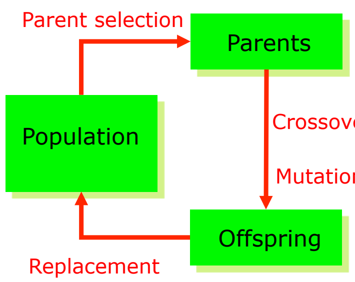
Write neatly and clearly.

Student Code Ethics

Students are expected to maintain the highest standards of academic integrity. Work that is not the students' own creation will receive no credit. Remember that you cannot give or receive unauthorized aid on any assignment, quiz, or exam. Students cannot use others' ideas and declare that they belong to them. Students are required to properly cite the original sources of the ideas and information used in their work.

Exercise 1 (3 points)
Which are the genetic operators of the Simple Genetic Algorithm? Explain how they work.

Solution
The main loop of the Simple Genetic Algorithm is



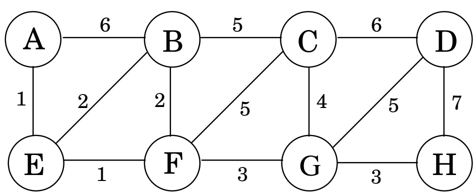
|Population| = |Parents| = |Offspring|

There are 3 genetic operators: selection, crossover, and mutation

- **Selection** picks individuals from Population with probability proportional to their fitness: fitter individuals (better solutions to the problem) have greater chance to be selected. Selection creates the set Parents, which possibly includes multiple copies of the best individuals.
- **Crossover** is then applied to Parents. Each individual in Parents has a probability p_c to be chosen. There are different types of crossover, see lecture 10 for details. Crossover has the goal of creating new (possibly better) individuals (solutions), by combining the genotypes of the parents.
- **Mutation** is then applied on single individuals. It changes the genotypes by flipping each bit with a very small probability. The goal is to maintain genetic diversity from one generation to the next.

The whole population is replaced at each iteration of the algorithm.

Exercise 2 (4 points)
The Prim's algorithm finds a minimum spanning tree (MST) on a connected graph. For the graph below, show the steps in executing Prim's algorithm. Draw the partial MST identified at each step.



In general, can a connected graph have more than one MST? If not, explain why; if yes, make an example.

Solution
The application of the Prim's algorithm to the given graph finds a MST including the edges in the following table. In the table, the edges are in the order of inclusion in the MST. The starting node of the algorithm is node A.

Edge included
AE
EF
BE
FG
GH
CG
GD

In general, a connected graph can have more than 1 MST. For example, another MST for the given graph is obtained by replacing edge BE with edge BF.

Exercise 3 (4 points)

- Briefly describe the gradient descent algorithm.
- Which is the function that the gradient descent algorithm optimizes when applied to linear regression?
- Is the gradient descent algorithm assured to find the global minimum for linear regression? Explain.
- What is feature scaling and why is it useful before running the gradient descent algorithm?

Solution

- Gradient descent is an iterative algorithm used to find the minimum of a differentiable function. For simplicity here we consider a function of 2 variables θ_0 and θ_1, but the algorithm can be generalized to *n* variables. Given a function $J(\theta_0,\theta_1)$, gradient descent finds $min_{\theta_0,\theta_1} J(\theta_0,\theta_1)$. Starting from a point of the function, e.g., $(\theta_0=0,\ \theta_1=0)$, the algorithm keep changing θ_0 and θ_1 to reduce $J(\theta_0,\theta_1)$, until hopefully ends up at a minimum. The change of the 2 variables is done in the "steepest" direction, indicated by the gradient of the function wrt the variables. The algorithm can be summarized as follows, where α (learning rate) regulates the "step" of the change.

$$
\begin{aligned}
&\text{repeat until convergence } \{ \\
&\quad \theta_j := \theta_j - \underbrace{\alpha}_{\text{Learning rate}} \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \qquad (\text{for } j = 0 \text{ and } j = 1) \\
&\}
\end{aligned}
$$

- In case of linear regression the function $J(\theta_0,\theta_1)$ is the cost function below.

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

- The cost function is convex, so the gradient descent algorithm is assured to converge to the minimum, with the proviso that the learning rate α is not too large.
- Feature scaling is adopted to speed up convergence of the algorithm.

Exercise 4 (1 point)

Explain what the big-O notation is and make an example of two functions *f* and *g* such that $f=O(g)$.

Solution

Let $f(n)$ and $g(n)$ be functions from positive integers to positive real numbers. We say $f=O(g)$ (which means that "f grows no faster than g") if there is a constant $c>0$ such that $f(n) \le c^* g(n)$. An example is the following: $f(n) = 2n+20$, $g(n) = n^2$, $f=O(g)$.

Exercise 5 (3 points)

Explain what the maximum flow problem is. The Ford-Fulkerson algorithm and the Edmonds-Karp algorithm were proposed to solve the maximum flow problem; explain the main advantage (in terms of complexity) of the Edmonds-Karp algorithm over the Ford-Fulkerson algorithm.

Solution

Given a network (a directed graph) $G=(V,E)$ the maximum flow problem is to find a feasible flow from the source of G ($o \in V$) to the sink of G ($t \in V$) having the maximum value. Flow through an edge is allowed only in the direction indicated by the arrow and the maximum amount of flow through an edge is given by the edge capacity. The maximum flow is measured either as the amount of flow leaving the source *o* or as the amount of flow entering in the sink *t*.

The key idea behind the Ford-Fulkerson algorithm is that as long as there is a path from the source to the sink, with available capacity on all edges in the path, we send flow along it. A path with available capacity is called an augmenting path; in general at each step of the

algorithm there are several alternative augmenting paths that can be used to increase the flow. The runtime of Ford-Fulkerson is bounded by $O(|E|*f)$, where $|E|$ is the number of edges in the graph and f is the maximum flow. If the number of edges is small but the maximum flow is very high, the running time can be also high (see lecture 9 for an example).
The Edmonds-Karp algorithm uses a heuristics to select the augmenting path, i.e., it selects the shortest augmenting path. This bounds the running time to $O(|V|*|E|^2)$ and makes it independent from the maximum possible flow.

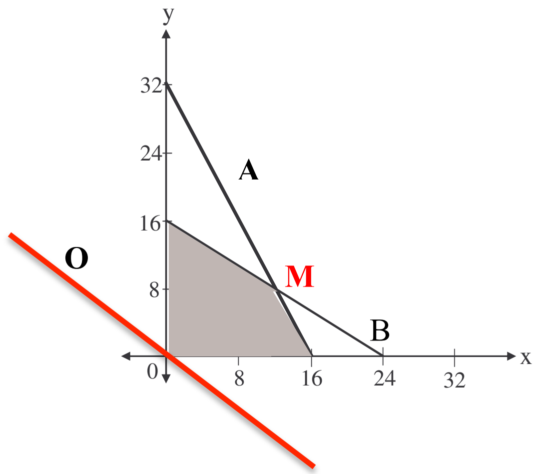
<u>Exercise 6</u> (3 points)
A small business enterprise makes dresses and trousers. To make a dress requires 1/2 hour of cutting and 20 minutes of sewing. To make a pair of trousers requires 15 minutes of cutting and 1/2 hour of sewing. The profit on a dress is 40 € and on a pair of trousers is 50 €. The business operates for a maximum of 8 hours per day. Determine how many dresses and trousers should be made to maximize the daily profit and what the maximum profit is.

1. Formulate a linear programming model for this problem;
2. Use the graphical method to solve this model.

<u>Solution</u>
Let x be the number of dresses and y the number of trousers.

(A) Constraint on cutting time:	$1/2x + 1/4y \leq 8$	→	$2x+y\leq 32$
(B) Constraint of sewing time:	$1/3x+1/2y \leq 8$	→	$2x+3y\leq 48$
Trivial constraints:	$x\geq 0,\ y\geq 0$		
(O) Objective function:	max $40x+50y$		

Feasible region:



The line corresponding to all the pairs (x,y) for which the objective is equal to 0 is denoted by O. Given the slope of this line, we can visually determine that the maximum of the objective function is the vertex of the feasible region indicated by M. The coordinates of M can be found calculating the intersection of lines A and B; M=(12, 8). The value of the objective function in M is 880.