**CS4234: Optimization Algorithms** Oct. 22, 2015

# Tutorial Week 10

## Gradient Descent / Simulated Annealing Exercises

**Problem 1.** (Gradient Descent) Assume you have the following function of three variables:

$$f(x, y, z) = 3x^3 + e^{2y} + 2xy^2$$

Execute one iteration of gradient descent, starting at the point:

$$(x, y, z) = (10, \ln 10, 10)$$

For this iteration, set $\alpha = 0.01$. After one update, what is the value of $(x, y, z)$?

**Problem 2.** (Simulated Annealing) Imagine you are solving the Traveling Salesman Problem on 5 points in the Euclidean plane (and using Euclidean distance to measure distance):

$$\begin{aligned} A &= (0, 0) \\ B &= (10, 0) \\ C &= (20, 0) \\ D &= (20, 10) \\ E &= (10, 10) \end{aligned}$$

You currently have the following candidate tour:

$$(A, E, B, C, D)$$

(where you return to $A$ after $D$ at the end of the tour). You are considering an optimization heuristic where you can swap any two consecutive cities on the tour. For example, beginning from the tour above, you may choose to adopt tour $(A, B, E, C, D)$ but not tour $(A, C, E, B, D)$. You hope that by repeating a sequence of these swaps, you will eventually find a good tour. (*Note: this is different from the 2-OPT heuristic mentioned previously.*).

You decide to run simulated annealing. Beginning from the tour $(A, E, B, C, D)$, perform one iteration of simulated annealing where $T = 5$. (Notice that we are trying to *minimize* the tour length.)

- How many *neighboring* tours are there?
- For two of the possible outcomes, calculate the probability that the algorithm selects that outcome as the next state.

**Problem 3.** Assume you are given an undirected graph $G = (V, E)$ with $n$ nodes and $m$ edges. Your goal is to orient each edge, i.e., for every undirected edge $e = (u, v) \in E$, output either $(u \to v)$ or $(v \to u)$. Define the degree $\Delta(v)$ in the output graph to be out-degree of node $v$, i.e., $|\{(v \to u) | u \in V\}|$. For a given orientation, define $\Delta$ to be the maximum degree of any node. Your goal is to find an orientation that minimizes $\Delta$, i.e., minimizes the maximum out-degree of any node.

Assume we have a candidate (arbitrary) orientation. Consider the following iterative step:

- Choose a node $u$ with degree $d$.
- Search for a directed path $P$ from $u$ to a node $v$ with out-degree $\Delta(v) \leq d - 2$.
- If there exists such a path $P$, reverse all the nodes on path $P$. Otherwise, do nothing.

Repeat this algorithm until no further changes are possible, i.e., there is no directed path from a node with higher degree $d$ to a node with lower degree $\leq d - 2$.

Prove that the algorithm finds an optimal orientation. (Hint: Take a node $u$ with maximum degree in the oriented graph; look at the set of nodes $V'$ that are reachable from $u$ on directed edges. What do we know about these nodes? What is the out-degree of OPT on these nodes?)

**Fun facts.** For every planar graph, there is an orientation with maximum out-degree 3. For sparse graphs, there are also good bounds: assume that for every subset $V' \subseteq V$, the set of edges $|E[V']| \leq k(|V'| - 1)$; then there is an orientation with maximum out-degree $k$. (The notation $E[V']$ refers to the set of edges that connect nodes in $V'$.) Notice that if for some subset $V'$, $|E[V']| > k|V'|$, then it is clearly impossible to have an orientation with maximum out-degree $k$. This is closely related to the idea of *arboricity* and can be proven using a famous theorem by Nash-Williams.