Advanced Algorithms 2011. Exam Answers

1. No. By definition, an FPTAS yields an approximation ratio arbitrary close to 1 for any fixed instance. This is not the case for the greedy algorithm for Load Balancing.

2. Let S be a solution with optimal perimeter, and let p and q be points in S with maximum distance $|pq|$. Our approximation algorithm takes also p and chooses, for each color, the point closest to p. All these points are inside the circle of radius $|pq|$ with center p (since S is inside this circle, thus points of each color exist in S). Hence our solution has perimeter at most $2\pi|pq|$, while S has perimeter at least $2|pq|$.

3. If not, then some edge e is uncovered by S (and M), but then the algorithm would continue, by adding e to M.

4. Every vertex cover must contain, in particular, at least one node of every $e \in M$. Since no two edges of M share a node, at least $|M|$ nodes are needed. The algorithm selects $2|M|$ nodes.

5. Not necessarily. The simplest counterexample is a path of three edges. The algorithm may select the middle edge, thus block the others. But the outer edges build a matching of size 2.

6. We construct an instance of Set Cover as follows. The ground set has m elements v_j. For each $a \in A$ we create a set containing all v_j for which $a \in B_j$. The weight of this set is the weight of a. Obviously the reduction needs polynomial time, and the two problems are equivalent.

7. Wrong. One possible motivation: The number of nodes in the two parts is not specified, i.e., the number of pairs of nodes from the two parts is not fixed, hence the maximization problem is not turned into a minimization problem in this trivial way. (Of course, this argument does not yet rule out the possibility of other reductions.)

8. A bulb works with probability $1 - p$. Due to independence, all m bulbs in a group work with probability $(1 - p)^m$. Hence we observe an expected number of $k(1 - (1 - p)^m)$ defective groups. The actual number of observed defectives could be used to estimate p, using this formula.

9. Put every node into part 1 or part 2, independently with probability 1/2. Then every edge is in the cut with probability 1/2. Hence an expected number of $m/2$ edges is in the cut. It follows the existence of a cut with at least $m/2$ edges. We find one by repeating the above algorithm until success.

10. Yes (if you proposed this algorithm). The reasoning is the same as for fulfilling $7k/8$ clauses in a 3-SAT formula.

11. The simple solution is: Use an arbitrary order! If more than $m/2$ edges have the wrong direction, take the reverse order.

12. No, at least not immediately. Chernoff bounds apply to sums of independent random variables. But the steps of Quicksort (whose running times we would consider as random variables that together make up the total running time) are not independent.

13. Assign an arbitrary color to the first node. Then, always color a neighbor of a node that is already colored. Only 2 colors, rather than 3, are permitted. If no uncolored node has a colored neighbor, we can again assign an arbitrary color to some node. The branching number is never larger than 2.

14. If t is our time budget, we have $3^N = t$, hence $N = \log_3 t$. Let M be the size manageable by the faster algorithm. That is, $2^M = t$, hence $M = \log_2 t = \log_2 3 \cdot \log_3 t = \log_2 3 \cdot N$.

15. First apply any 2-approximation algorithm for Vertex Cover. Let c' be the obtained solution size. If $c' \leq 2k$, apply an $O^*(1.47^k)$ time exact algorithm. If $c' > 2k$, we know that $c \geq c'/2 > k$, and we can stop. The algorithm behaves as disired: If $c \leq 2k$, it runs in $O^*(1.47^k)$ time, and if $c > 2k$ then also $c' \geq c > 2k$, hence only the polynomial time for the approximation is used.

16. This cannot work, unless P=NP. We could iterate the alleged algorithm (with the same k, and G' being the next G) less than n times, and eventually decide in polynomial time whether the original graph has a vertex cover of size k. (This shows that any polynomial-time kernelization must in general stop at a kernel size strictly larger than k.)