Polynomial Computations (exercises)

Let $\mathfrak{M}$ be a deterministic Turing machine. For an input word x by $t_{\mathfrak{M}}(x)$ we denote the number of steps $\mathfrak{M}$ performs when running on x (if it never stops, $t_{\mathfrak{M}}(x) = \infty$). By $T_{\mathfrak{M}}(n)$ we denote the maximal value of $t_{\mathfrak{M}}(x)$ where x ranges over words of length n:

$$T_{\mathfrak{M}}(n) = \max\{t_{\mathfrak{M}}(x) \mid |x| = n\}.$$

1. Suppose that $\mathfrak{M}$ has m states, k letters in its internal alphabet, and uses at most $s(x)$ memory cells when running on input x. Give an upper bound for $t_{\mathfrak{M}}(x)$, provided it is not ∞ (that is, $\mathfrak{M}$ stops on x).

2. Consider a Turing machine $\mathfrak{M}_2$ with two tapes. At each step, it operates on each tape. Show that there exists a one-tape Turing machine $\mathfrak{M}$ that computes the same function as $\mathfrak{M}_2$. Give an upper bound for $T_{\mathfrak{M}}(n)$ in terms of $T_{\mathfrak{M}_2}(n)$. Do the same for the more general case of a k-tape machine $\mathfrak{M}_k$.

3. Does there exist an polynomial time algorithm for checking satisfiability of Boolean formulae in Disjunctive Normal Form?

4. Does there exist a polynomial time algorithm which checks whether a given graph is bipartite?

5. Does there exist a polynomial time algorithm which takes a graph and

   (a) determines whether it has a Euler cycle;

   (b) if the answer is "yes,", returns such a cycle?

6. Does there exist a polynomial time algorithm which takes a bipartite graph and

   (a) determines whether it has a perfect matching;

   (b) if the answer is "yes," returns one of such matchings?