Development knowledge-based systems led to appearance of various approaches to formalizing uncertain knowledge and reasoning. The most well-known example is Bayes' method of combining evidence. A different kind of uncertain knowledge and reasoning is formalized in probabilistic logic proposed by N. Nilsson in [1]. This approach is an extension of ordinary logic to the case when we represent knowledge in the form of an axiomatic theory but are not confident in truth of the axioms. Ordinary logic tell us what sentences are true if the axioms of our theory are true. In probabilistic logic we can talk about the probability of a sentence, i.e. the probability that it is true. The axioms of a probabilistic theory are true with some probabilities. Knowing these probabilities (or intervals in which they lie), we can find bounds for probabilities of other sentences.

The semantics of probabilistic logic is defined in terms of first order logic and probability theory (see [1] and more detailed expositions in [2],[3]). Within its framework there are different reasoning methods. They are based on either solving systems of linear equations or appropriate axiomatizations for reasoning about probabilities ([4],[5]). Here we do not discuss this material. The present paper is concerned with an approach which is based on the idea of probabilistic logic and has the following features in comparison with Nilsson's case:

(i) the approach is defined by means of the notion of an arbitrary
inference engine, and so, can be applied to as logical systems with the standard first order logic semantics as other deductive systems, for example, Prolog:

(ii) probabilities of axioms determine probabilities of other sentences uniquely;

(iii) axioms are to be mutually independent (in some precise sense).

Necessary definitions are given in section 2. We define what we mean by an algorithm for probabilistic inference. In sections 3 and 4 we discuss such algorithms, mainly, from the point of view of their complexity. In particular, we describe one of them which is based on Monte-Carlo technique. The last section is concerned with algorithms for probabilistic reasoning in logic programming.

2. Probabilistic inference

Let a set of words, called sentences, be given. This set contains together with each sentence S another which is called its negation and is denoted by \( \neg S \). By a theory we mean a finite set of sentences called axioms of the theory. By an inference engine we mean any algorithm I such that:

(i) I is applied to an input consisting of a theory T and a sentence S;

(ii) I halts on all inputs and returns either 1 or 0, in case of \( I(T, S)=1 \) we say that T implies S or S is inferred from T (by I);

(iii) \( I(T, S) \) is 1 for every axiom S of T.

A theory T is said to be consistent (relatively an inference engine I) if there exists no sentence S such that both S and \( \neg S \) are inferred from T.

Let T be a theory with axioms \( S_1, \ldots, S_n \). In ordinary logic the axioms are considered to be true in a domain described by this theory. In our case we are not confident in truth of the axioms, some of them can be false. Therefore, we can talk about \( 2^n \) possible theories with axiom sets of the form \( S'_1, \ldots, S'_n \) where \( S'_i \) denotes either \( S_i \) or \( \neg S_i \). Only one of this theories is true actually but we do not know which of them.