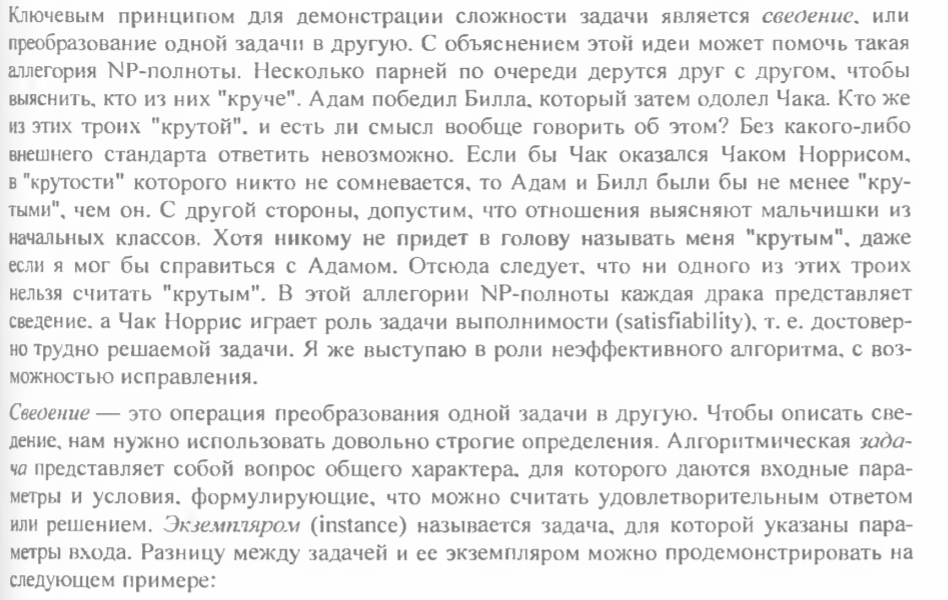
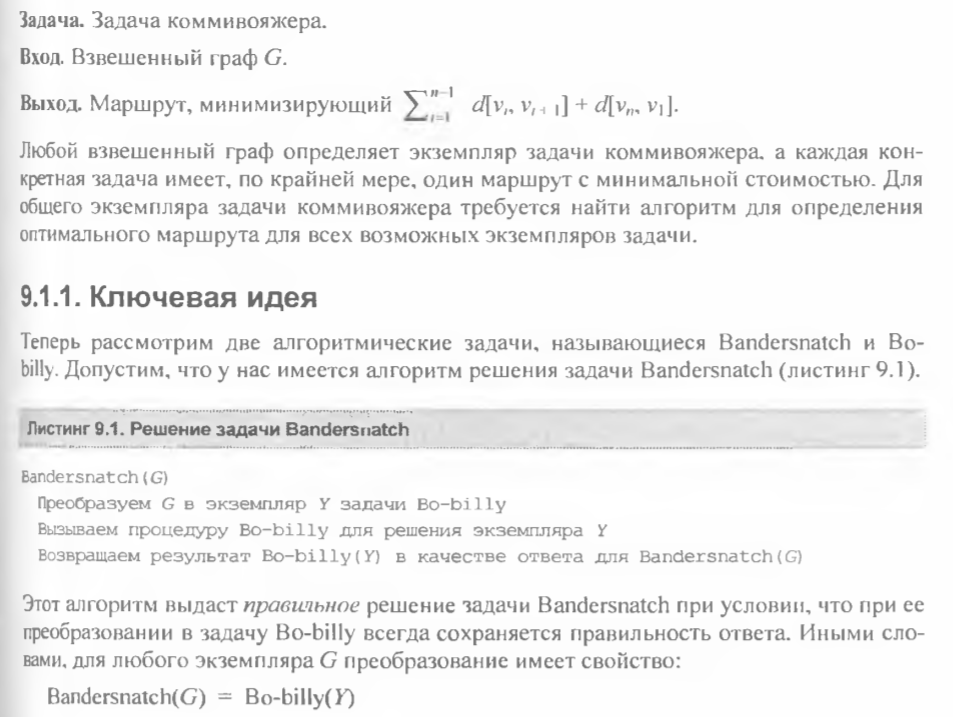
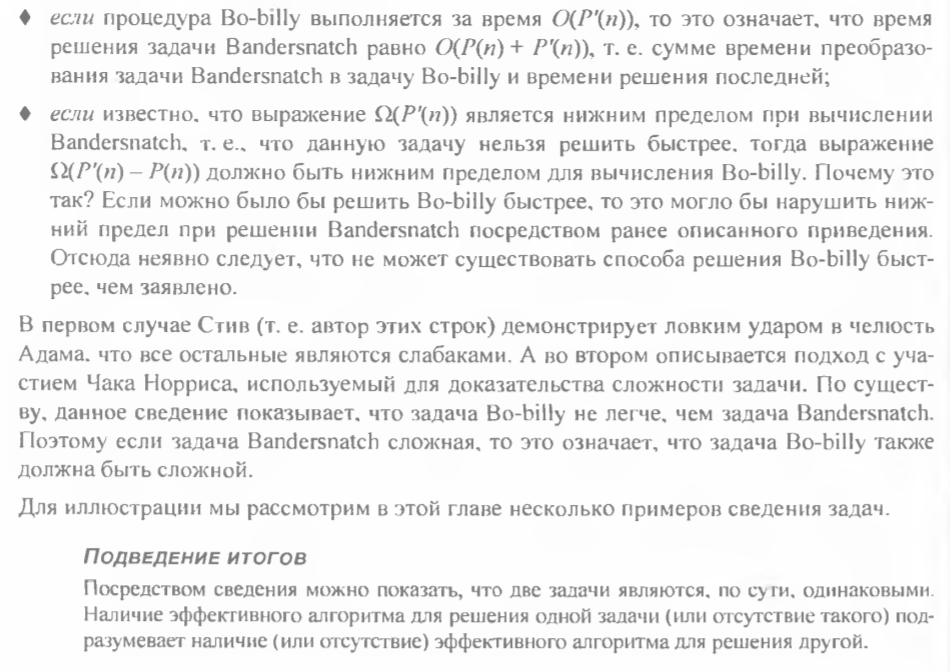
**ГЛАВА 9 Труднорешаемые задачи и аппроксимирующие алгоритмы**

**9.1. Сведение задач**

****

****



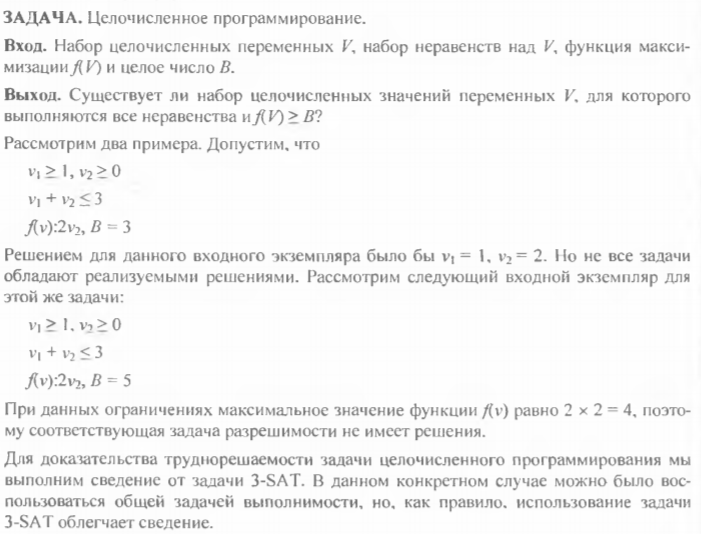
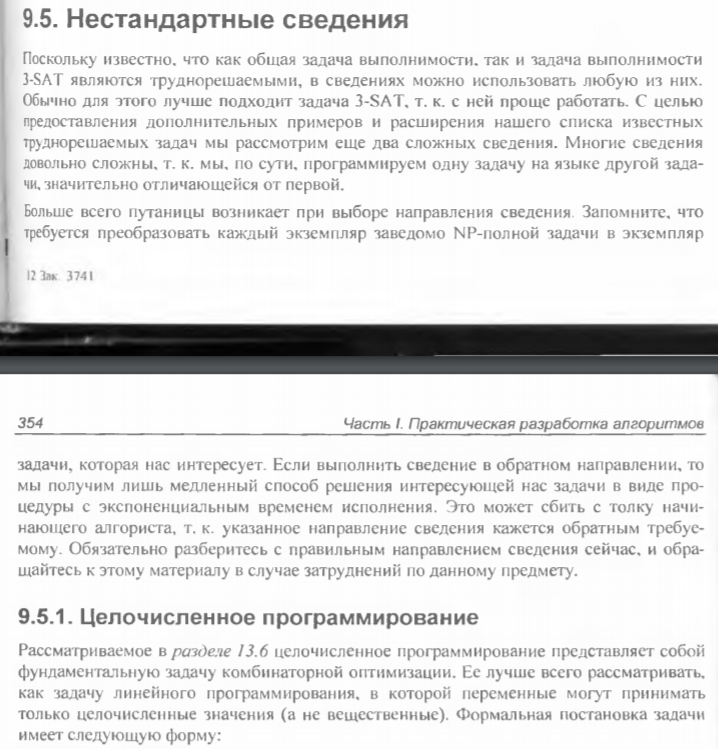
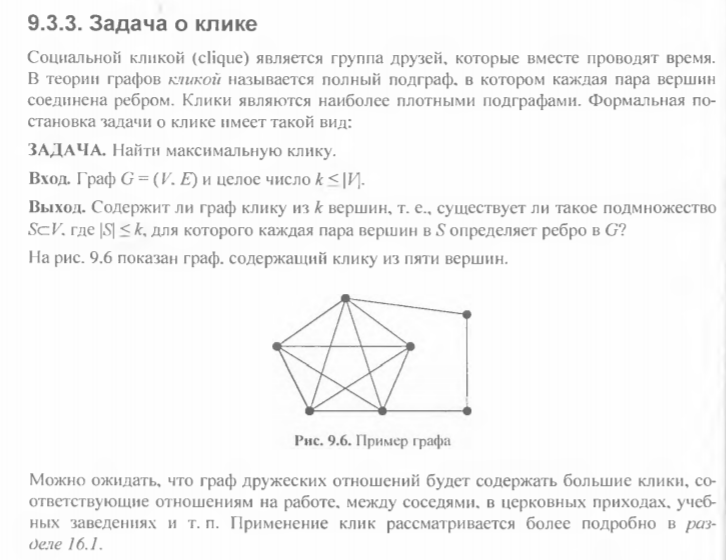
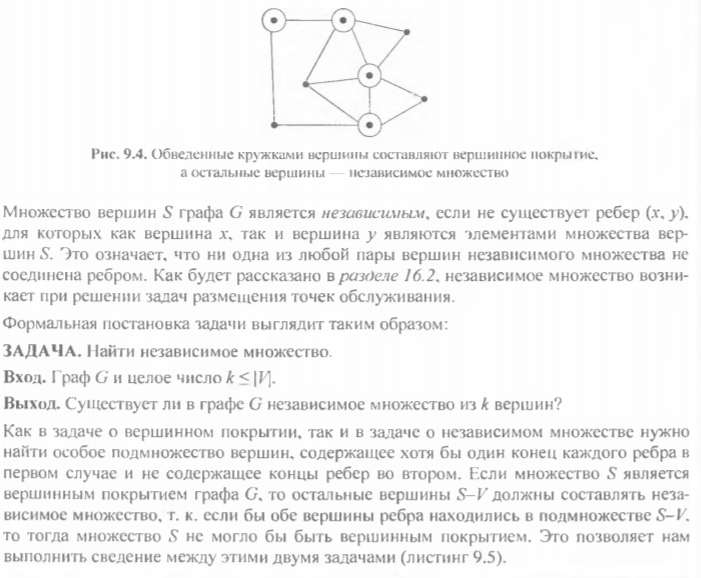
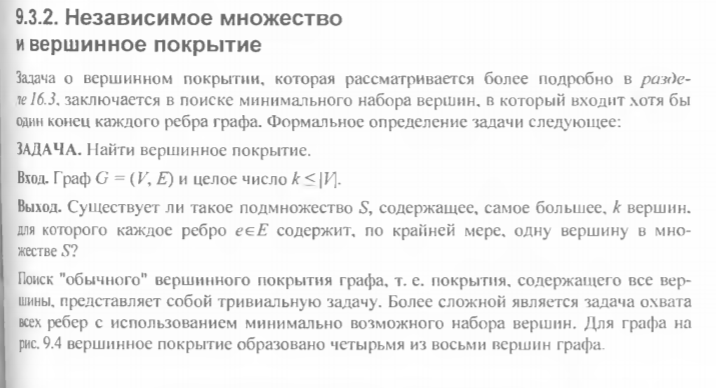
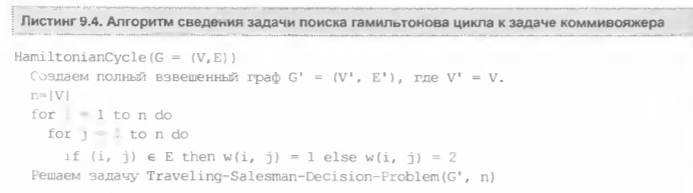
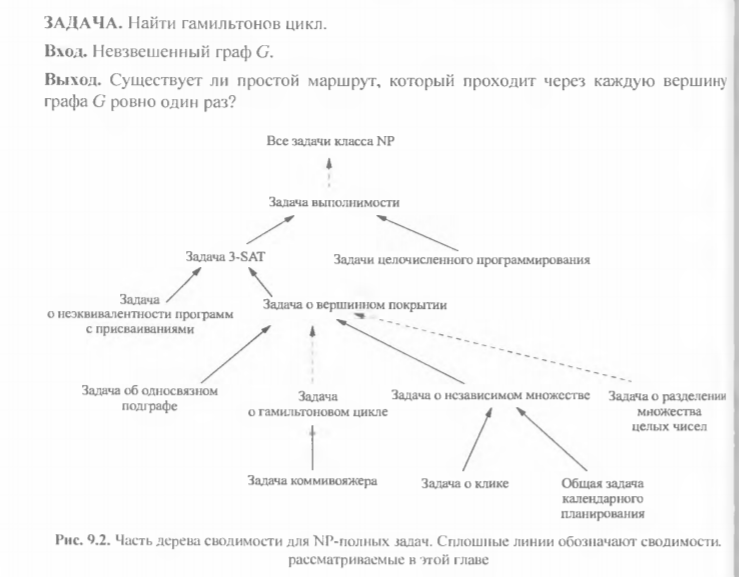
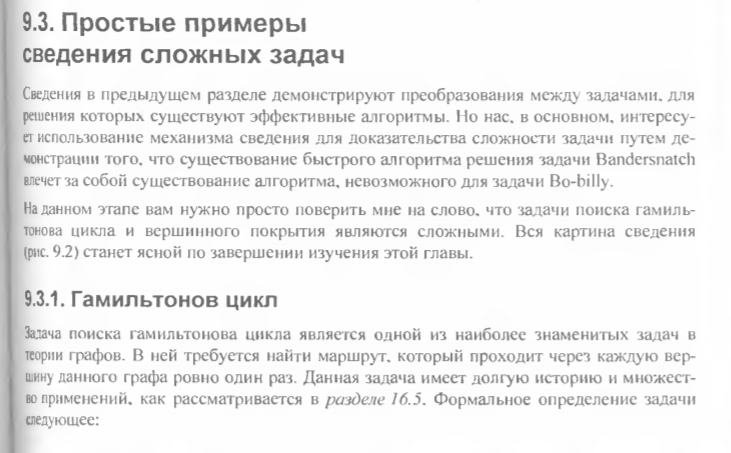
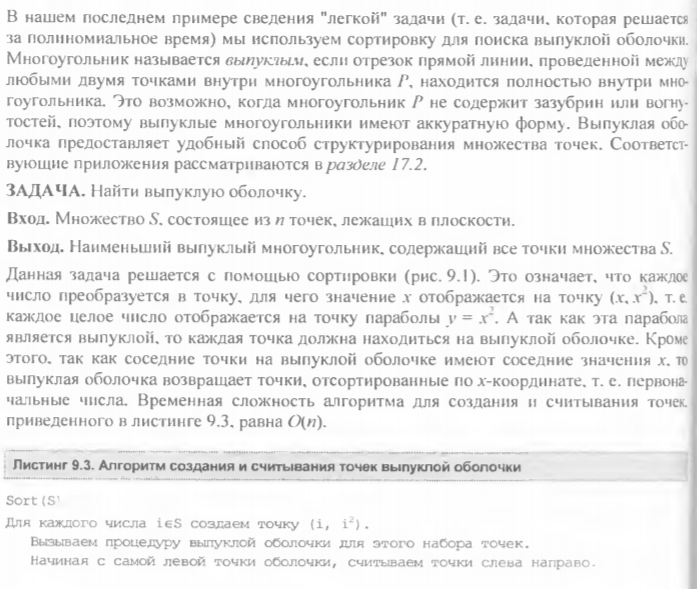
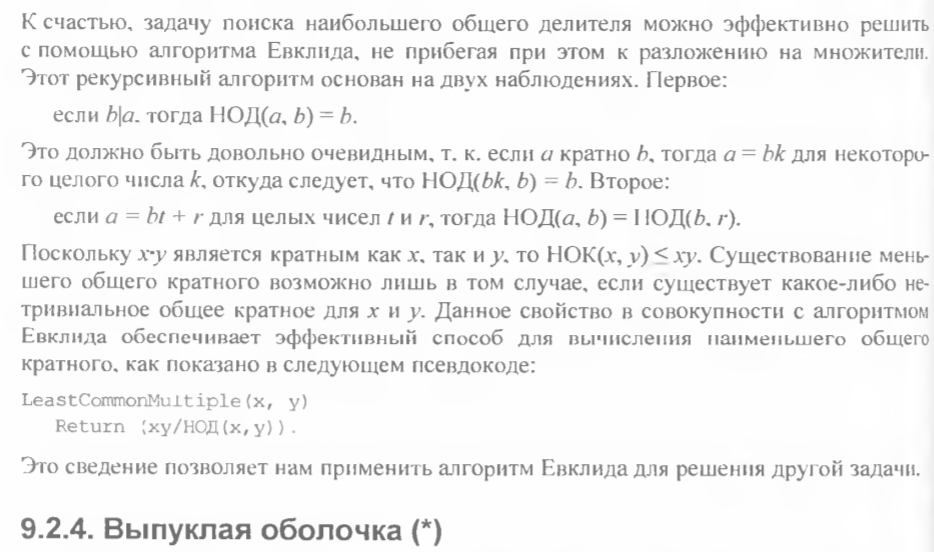
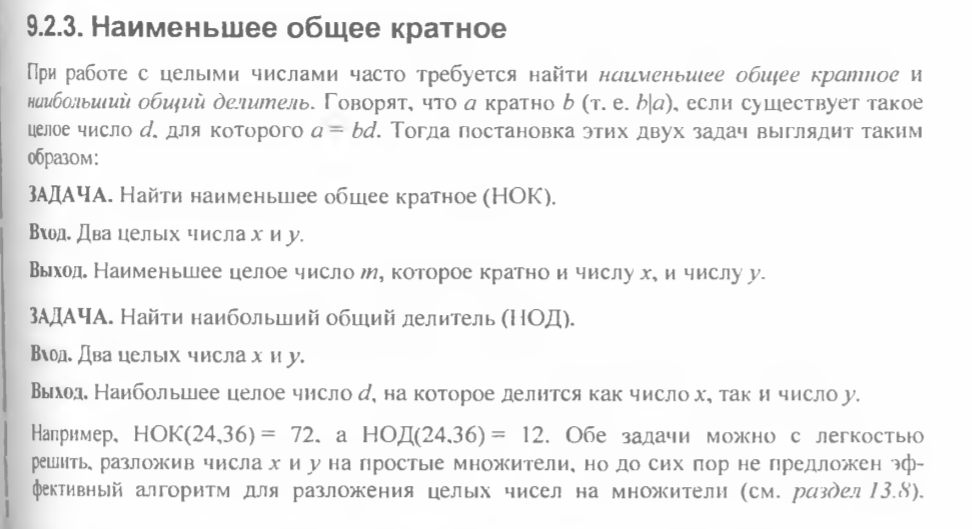
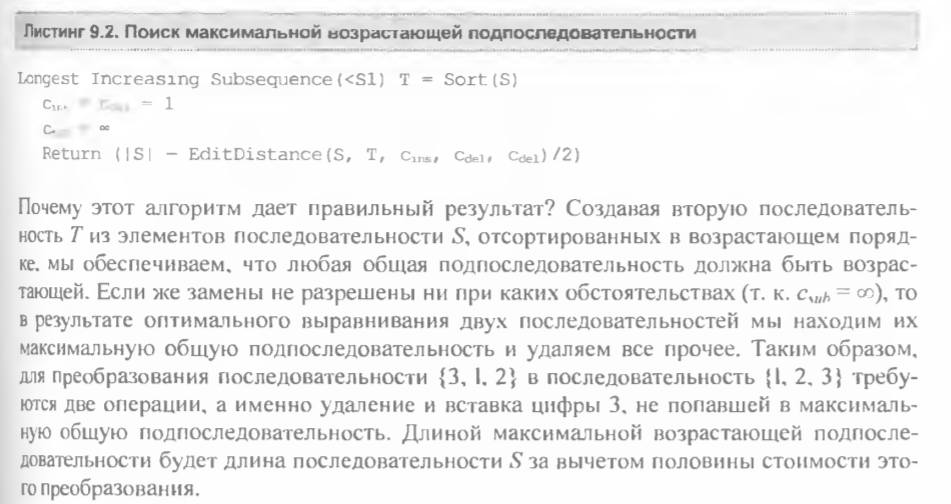
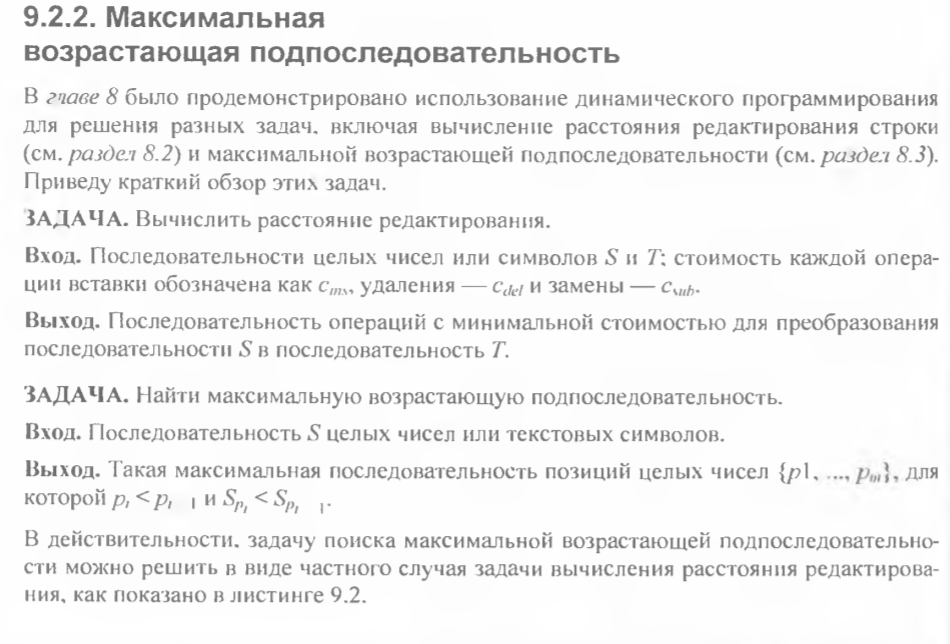
**9.1.2. Задачи разрешимости**

При сведении задача одного типа преобразуется в задачу другого типа таким образом, что ответы для всех экземпляров задачи являются идентичными. Задачи отличаются друг от друга диапазоном или типом возможных ответов. Решение задачи коммивояжера состоит из перестановки вершин, в то время как решения других задач возвращаются в виде чисел, диапазон которых может быть ограничен положительными или целыми числами. Диапазон решений наиболее интересного класса задач ограничен значениями ИСТИНА или ЛОЖЬ. Эти задачи называются задачами разрешимости (decision problems). Удобно сводить одну задачу разрешимости к другой, т. к. допустимыми ответами как для начальной, так для конечной задачи являются только ИСТИНА или ЛОЖЬ.

**9.2. Сведение для создания новых алгоритмов**

На кухне сидят инженер и апгорист. Алгорист просит инженера вскипятить воды для чая. Инженер встает со своего стула, берет со стола чайник, наливает в него воды, ставит его на плиту, зажигает горелку, ждет, пока вода закипит, после чего выключает горелку. Некоторое время спустя инженер просит апгориста вскипятить еще воды. Тот поднимается со стула, берет чайник с плиты, ставит на стол и снова садится. — Сделано, — говорит он. — Я свел задачу, требующую решения, к уже решенной задаче.

Пример со сведением задачи кипячения воды иллюстрирует достойный способ создания новых алгоритмов из старых. Если входные данные для задачи, которую мы хотим решить, можно преобразовать во входные данные для задачи, которую мы знаем, как решить, то процедура преобразования и известное решение образуют алгоритм для решения нашей задачи.



**9.6. Искусство доказательства сложности**

Доказательство сложности задач требует определенного мастерства. Однако, приобретя навык, вы обнаружите, что выполнение сведений может быть на удивление простым процессом. Малоизвестной особенностью доказательств NP-полноты является тот факт, что их легче создать, чем объяснить, аналогично тому, как часто бывает легче заново написать код, чем разбираться в старом. Умение оценить вероятность того, что задача трудна для решения, требует опыта. Возможно. что самым быстрым способом набраться такого опыта будет внимательное изучение примеров в каталоге задач. Незначительное изменение формулировки задачи может сделать полиномиальную задачу NP-полной. Задача поиска кратчайшего пути в графе является легкой, в то время как задача поиска самого длинного пути в графе является сложной. Задача построения маршрута, который проходит по всем ребрам графа ровно один раз (цикл Эйлера), является легкой, а вот задача построения маршрута, который проходит через каждую вершину графа (гамильтонов цикл) является сложной. Если вы подозреваете, что задача является NP-полной, сначала проверьте, не содержится ли она в книге [GJ79], в которой перечислены несколько сотен известных NPполных задач. Велика вероятность, что вы найдете свою задачу в этой книге.

♦ Примите меры к тому, чтобы виши исходная задача была как можно проще (т. е. максимально ограничена).

♦ Сделайте целевую задачу максимально сложной.

♦ Выбирайте исходную задачу на основании веских аргументов.