Finally, we show that the problem of finding the edge-chromatic index of a graph is *NPC*. In fact, we specifically prove the stronger result that it is *NP*-complete to determine whether or not the edge-chromatic index of a 3-regular graph is 3 or 4. Of course, the edge-chromatic index cannot be less than 3 and by Vizing's theorem (chapter 7) it cannot exceed 4.

Cubic graph edge-colouring (CGEC)

Instance: A 3-regular graph G.

Question: Does there exist a (proper, edge-)colouring of G using three colours?

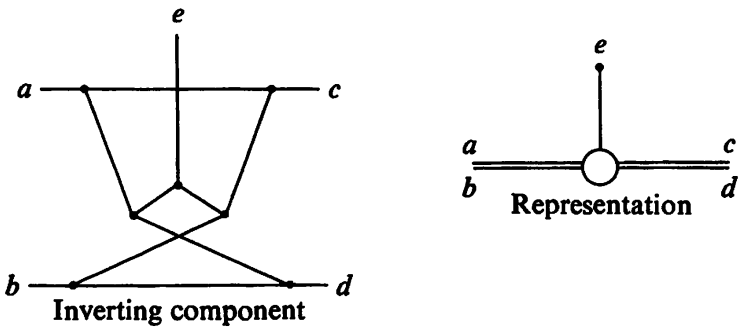
Theorem 8.12. *CGEC* is *NPC*.

Proof. Clearly $CGEC \propto NP$. We complete the proof by outlining a transformation: $3SAT \propto CGEC$. In other words, we show how to construct, from an instance I of $3SAT$, a 3-regular graph G which is 3-colourable if and only if I is satisfiable.

G is constructed from a number of components each of which is designed to perform a specific task. These components are connected by pairs of edges such that, in a 3-edge-colouring of G, a pair represents the value *true* if both edges are identically coloured and if they are differently coloured then they represent the value *false*.

A key component of G is the *inverting component* shown in figure 8.9 along with its symbolic representation.

Fig. 8.9



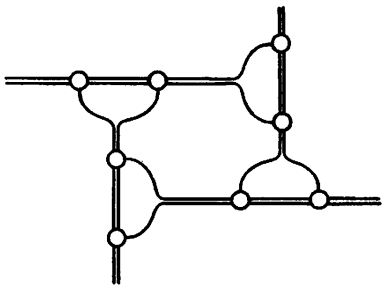
Lemma 8.3. In a 3-edge-colouring of an inverting component, the edges in one of the pairs of edges (a, b) or (c, d) are similarly coloured, whilst the remaining three labelled edges have distinct colours.

Proof. Like the previous two lemmas we leave this an exercise. However, the comment in exercise 8.9 may be of use. ■

If for the inverting component we look upon the pair of edges (a, b) as input and the pair (c, d) as output, then this component may be regarded as turning a representation of **true** into one of **false** and vice-versa.

We next describe a *variable-setting component* of G. Such a component exists for each variable v_i of the instance of $3SAT$. An example of such a component is shown in figure 8.10. This has four pairs of output edges. In general, however, there should be as many output pairs as there are appearances of v_i or its complement $\bar{v}_i$ in the clauses of the instance of $3SAT$. If there are x such appearances, then we can make, in an obvious way, a variable-setting component from $2x$ inverting components.

Fig. 8.10. A variable-setting component.

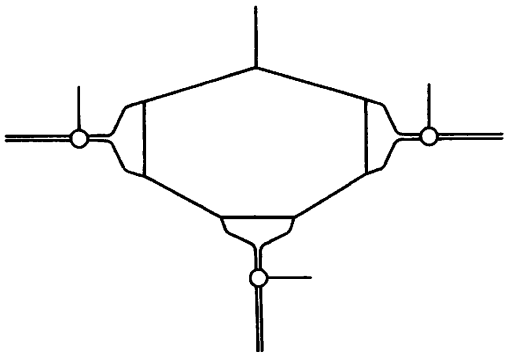


Lemma 8.4. In any 3-edge-colouring of a variable-setting component, all output pairs are forced to represent the same value, **true** or **false**.

Proof. Again, this is straightforward and is left as an exercise. ■

Finally we describe a *satisfaction-testing component* of G. Such a component is shown in figure 8.11. The required property of this component is embodied in the following lemma.

Fig. 8.11. A satisfaction-testing component.



Lemma 8.5. A 3-edge-colouring of a satisfaction-testing component is possible if and only if at least one of the input pairs of edges represents the value **true**.

Proof. This is straightforward and is left as an exercise. ■

Given an instance I of $3SAT$ we now show how to construct the 3-regular graph G which is 3-colourable if and only if I is satisfiable. For each v_i, we construct a variable-setting component V_i which has an output pair of edges for each appearance of the variable v_i or its complement $\bar{v}_i$ amongst the clauses of I. For each clause c_j of I we have a satisfaction-testing component C_j. Let $l_{j,k}$ be the kth literal of c_j. If $l_{j,k}$ is the variable v_i then identify the kth input pair of C_j with one of the output pairs of V_i. Otherwise, if $l_{j,k}$ is $\bar{v}_i$ then place an inverting component between the kth input pair of C_j and the output pair of V_i. Let H be the graph resulting from this construction. H will have some unmatched connecting edges from the C_j. In order to construct the 3-regular graph G, we simply take two copies of H and join them together by identifying the unmatched edges.

It is easy to see that G can be constructed in polynomial time. We complete the proof by noting that the properties of the components, as described in lemmas 8.3, 8.4 and 8.5, ensure that G can be 3-edge-coloured if and only if the instance I of $3SAT$ is satisfiable. ■

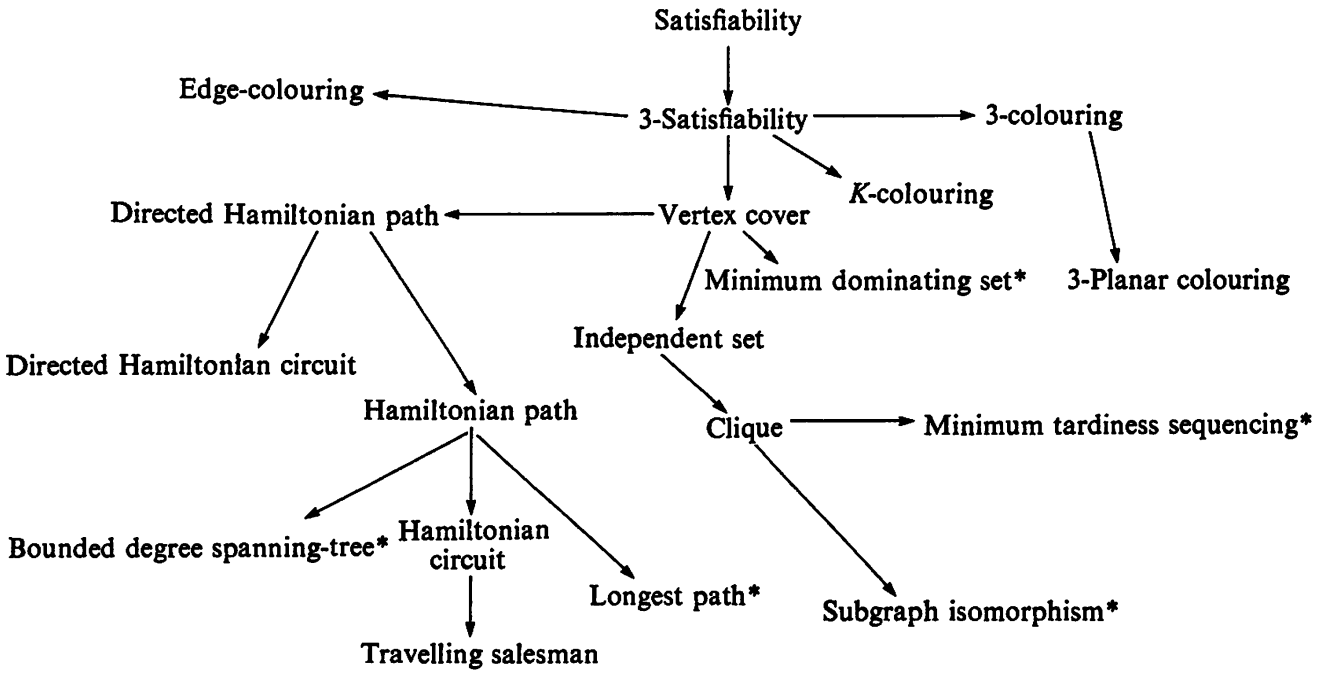
8.3 Concluding comments

Figure 8.12 shows the tree of transformations we have developed through the theorems of this chapter which, along with Cook's theorem, establishes some members of the class of *NP*-complete problems. If $P \neq NP$ then, of course, no member of this class has an efficient algorithmic solution. On the other hand, if an efficient solution is found for one, then there exists an efficient algorithm for any other. A great deal of effort has been fruitlessly expended in the search for efficient algorithms so that there is widespread belief that $P \neq NP$. Graph theory contains a large number of problems that are *NP*-complete and the reader is referred to Garey & Johnson[1] for the complete list.

The number of problems in figure 8.12 is relatively small, although it includes perhaps the best-known *NP*-complete graph problems and certainly the most important regarding material in this book. In this respect we might also have included problems of multicommodity flow,[9] maximum cuts[10] in networks and the Steiner tree problem.[11]

The establishment of *NP*-completeness for a problem need not be the final point in consideration of its time-complexity. We can proceed in

Fig. 8.12



* The asterisked problems are described in the exercise section at the end of the chapter