O algoritmo para construção da árvore Patricia é baseado no método de pesquisa digital, mas sem apresentar o inconveniente citado para o caso das tries. O problema de caminhos de uma só direção é eliminado por meio de uma solução simples e elegante: cada nó interno da árvore contém o índice do *bit* a ser testado para decidir qual ramo tomar. A Figura 5.11 apresenta a árvore Patricia gerada a partir das chaves B, C, H, J e Q apresentadas acima.

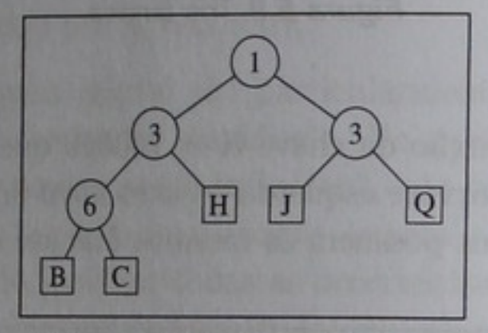

Figura 5.11 *Árvore Patricia.*

Para inserir a chave K = 100010 na árvore da Figura 5.11, a pesquisa inicia pela raiz e termina quando se chega ao nó externo contendo J. Os índices dos *bits* nas chaves estão ordenados da esquerda para a direita. Assim, o *bit* de índice 1 de K é 1, indicando a subárvore direita, e o *bit* de índice 3 indica a subárvore esquerda que, neste caso, é um nó externo. Isso significa que as chaves J e K mantêm o padrão de *bits* 1x0xxx, assim como qualquer outra chave que seguir este caminho de pesquisa. Um novo nó interno repõe o nó J, e este, juntamente com o nó K, serão os nós externos descendentes. O índice do novo nó interno é dado pelo primeiro *bit* diferente das duas chaves em questão, que é o *bit* de índice 5. Para determinar qual será o descendente esquerdo e o direito, é só verificar o valor do *bit* 5 de ambas as chaves, conforme mostrado na Figura 5.12.

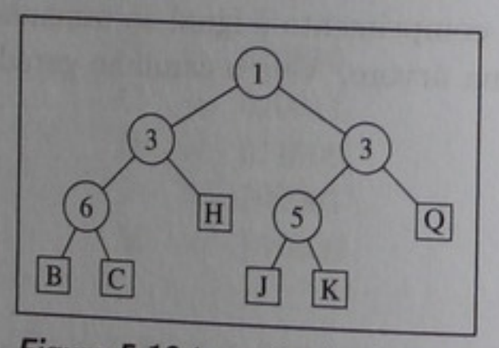

Figura 5.12 *Inserção da chave K.*

A inserção da chave W = 110110 ilustra outro aspecto. A pesquisa sem sucesso na árvore da Figura 5.13 é realizada de maneira análoga. Os *bits* das chaves K e W são comparados a partir do primeiro para determinar em qual índice eles diferem, sendo, nesse caso, os de índice 2. Assim, o ponto de inserção agora será no caminho de pesquisa entre os nós internos de índice 1 e 3. Cria-se aí um novo nó interno de índice 2, cujo descendente direito é um nó externo contendo W e cujo descendente esquerdo é a subárvore de raiz de índice 3, conforme ilustra a Figura 5.13.

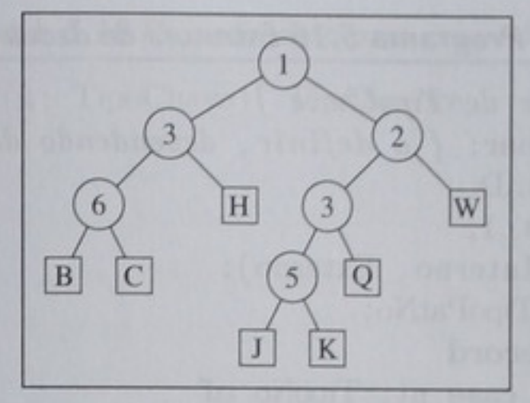

Figura 5.13 *Inserção da chave W.*

A implementação apresentada a seguir é derivada de Albuquerque e Ziviani (1985). O Programa 5.16 apresenta a definição da estrutura de dados utilizada na implementação do algoritmo. Os Programas 5.17, 5.18 e 5.19 apresentam algumas funções e procedimentos utilizados pelos algoritmos de pesquisa e inserção. O Programa 5.20 apresenta a implementação do algoritmo de pesquisa. O Programa 5.21 apresenta a implementação do algoritmo de inserção.

Cada chave k é inserida de acordo com os passos abaixo, partindo da raiz:

1. Se a subárvore corrente for vazia, então é criado um nó externo contendo a chave k (isso ocorre somente na inserção da primeira chave) e o algoritmo termina.
2. Se a subárvore corrente for simplesmente um nó externo, os *bits* da chave k são comparados, a partir do *bit* de índice imediatamente após o último índice da sequência de índices consecutivos do caminho de pesquisa, com os *bits* correspondentes da chave k' deste nó externo até encontrar um índice i cujos *bits* difiram. A comparação dos *bits* a partir do último índice consecutivo melhora consideravelmente o desempenho do algoritmo. Se todos forem iguais, a chave já se encontra na árvore e o algoritmo termina; senão, vai-se para o Passo 4.
3. Caso contrário, ou seja, se a raiz da subárvore corrente for um nó interno, vai-se para a subárvore indicada pelo *bit* da chave k de índice dado pelo nó corrente, de forma recursiva.
4. Depois são criados um nó interno e um nó externo: o primeiro contendo o índice i e o segundo, a chave k. A seguir, o nó interno é ligado ao externo pelo apontador de subárvore esquerda ou direita, dependendo se o *bit* de índice i da chave k seja 0 ou 1, respectivamente.
5. O caminho de inserção é percorrido novamente de baixo para cima, subindo com o par de nós criados no Passo 4 até chegar a um nó interno cujo índice seja menor que o índice i determinado no Passo 2. Este é o ponto de inserção e o par de nós é inserido.