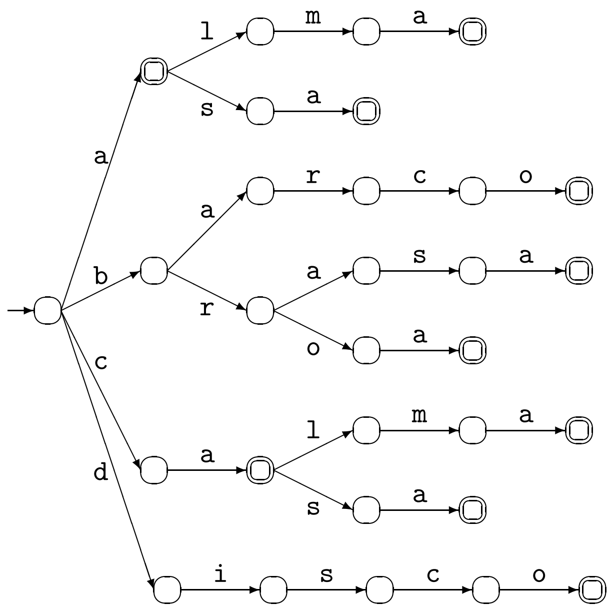

Figura 2.14 Um diagrama de estados do tipo árvore.

o AFD (ver o Exercício 14, no final da seção, página 89). Vê-se ainda que, se forem sendo introduzidas mais e mais palavras, a construção do AFD mais conciso vai se tornando mais e mais difícil, correndo-se o risco de obter um AFD incorreto ou com um número de estados maior que o necessário. Assim, é oportuno que exista o algoritmo de minimização da Figura 2.10, página 79: basta aplicar este algoritmo ao AFD do tipo árvore para obter o AFD mais conciso possível.

Assim, se uma linguagem é finita, existe um AFD que a reconhece cujo diagrama de estados simplificado não contém ciclos. Por outro lado, se um diagrama de estados simplificado de um AFD não contém ciclos, então a linguagem que tal AFD reconhece é finita. Assim, tem-se que: *uma linguagem é finita se, e somente se, existe algum AFD que a reconhece cujo diagrama de estados simplificado não tem ciclos*. Dizer isso é equivalente a dizer que *uma linguagem L é infinita se, e somente se,*

a) *não existe AFD que reconhece L; ou*

b) *o diagrama de estados simplificado de qualquer AFD que a reconhece tem ciclo.*

Ora, se um AFD reconhece uma linguagem infinita, é óbvio que seu diagrama de estados simplificado deve ter ciclo, pois uma linguagem infinita tem palavra de todo tamanho; em particular, há palavra de tamanho maior ou igual ao número de estados do AFD. E para reconhecer uma palavra de tamanho maior ou igual ao número de estados, deve-se passar por um ciclo.

Seja uma linguagem infinita. Como saber se existe ou não um AFD que a reconhece?

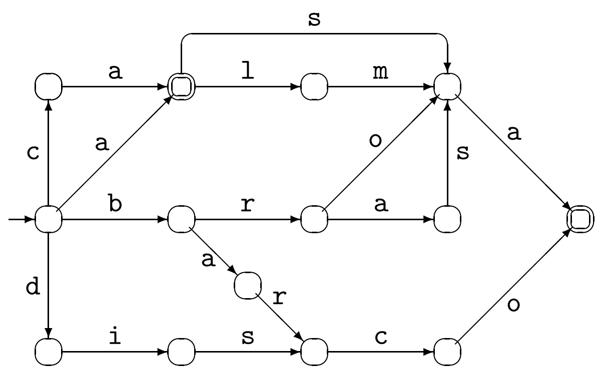

Figura 2.15 AFD mais conciso para o Exemplo 62.

Mais especificamente, como mostrar que *não existe* um AFD que reconhece uma linguagem quando todas as tentativas de construir um foram infrutíferas ou, melhor ainda, quando “se desconfia” que a linguagem tem uma estrutura um pouco complexa para ser reconhecível por AFD? Existem várias técnicas para isso, como será visto adiante. Uma delas tem como base o que foi descrito anteriormente, como mostra o exemplo a seguir.

Exemplo 63 Seja $L = \{\mathtt{a}^n\mathtt{b}^n \mid n \geq 0\}$. Como L é infinita, pode existir ou não um AFD para L. Suponha que exista um AFD M para L. Pelo exposto anteriormente, o diagrama de estados simplificado de M contém ciclo. Um ciclo é percorrido necessariamente para a computação correspondente ao reconhecimento de alguma palavra z de tamanho maior ou igual ao número de estados do AFD. Seja v uma subseqüência de z consumida ao se percorrer o ciclo (obviamente, $v \neq \lambda$). Nesse caso, $z = uvw$ para algum prefixo u e sufixo w. Ora, o ciclo pode ser percorrido quantas vezes se queira, inclusive 0, antes de consumir o sufixo w. Assim,

$$uv^iw \in L \text{ para todo } i \geq 0. \qquad (1)$$

Seja $z = \mathtt{a}^k\mathtt{b}^k$ para algum k tal que $|z|$ é maior ou igual ao número de estados de M. Então $uv^2w \notin L$, qualquer que seja v, pois:

- se v contém apenas $\mathtt{a}$s, $uv^2w = \mathtt{a}^{k+|v|}\mathtt{b}^k$;
- se $v = \mathtt{a}^i\mathtt{b}^j$ para $1 \leq i, j \leq k$, $uv^2w = \mathtt{a}^{k-i}(\mathtt{a}^i\mathtt{b}^j)^2\mathtt{b}^{k-j} = \mathtt{a}^k\mathtt{b}^j\mathtt{a}^i\mathtt{b}^k$; e
- se v contém apenas $\mathtt{b}$s, $uv^2w = \mathtt{a}^k\mathtt{b}^{k+|v|}$.

Isso contradiz a afirmativa (1). Observe, então, que a existência de ciclo implicaria o reconhecimento de palavras que não pertencem a L. Logo, não existe AFD para L. □

A técnica empregada no Exemplo 63 pode ser utilizada para mostrar que várias outras linguagens não podem ser reconhecidas por AFDs. Se, ao usar essa mesma