[0060] A recursive process is then invoked to identify character sequences that satisfy predefined usage frequency criteria. The recursive process starts with identifying a grandchild node (regardless of whether the grandchild node corresponds to a complete or partial word) of the input sequence node whose weight is sufficient to meet a predefined criterion for promotion to be a child node of the input sequence node. In some embodiments, the recursive process starts with identifying a grandchild node of the input sequence node whose weight is greater than mw_i, if any. If there are more than one grandchildren meeting this criterion, then it does not matter which one is chosen. In some other embodiments, the recursive process starts with identifying the highest-weight grandchild node of the input sequence node whose weight is exceeded by fewer than N child node weights, if any. If there is a grandchild node that meets the predefined criterion for promotion (**308**—yes), that grandchild node is promoted to be a child of the input sequence node (**310**). The grandchild node and its subtree are moved so that they descend directly from the input sequence node. The weight of the grandchild node is then subtracted from the weight of the node that is the original parent of the grandchild node (**312**), because the grandchild node and its subtree are no longer descendents of the original parent. If the updated weight of the original parent node is no longer greater than mwi (**314**—no), then the original parent node and its subtree are removed from further consideration (**316**). In either case (**314**—yes or no), the operations starting at **308** are then reapplied to the new character sequence tree with its new weights, and this continues until there is no longer a grandchild that meets the predefined criterion for promotion (**308**—no). Note that after a grandchild gets promoted, its children are added to the set of grandchildren of the input sequence node, and they too will be tested for promotion the next time around.

[0061] When there are no longer any grandchildren nodes of the input sequence node that meet the predefined criterion for promotion (**308**—no), the children of the input sequence node are sorted by their weights (**318**). One or more of the character sequences (which may include partial words, complete words, and/or phrases with two or more words) corresponding to the highest weighted children nodes are presented to the user (**320**). The character sequences are displayed on the display of the device. In some embodiments, up to a predefined number of the characters sequences are displayed, with the highest weighted character sequences in the weight-sorted order having the highest priority for display. The displayed character sequences may include partial words, complete words, and/or phrases with two or more words. When one or more grandchildren nodes have been promoted, and at least one of the promoted nodes is among the highest weighted children, one or more of the displayed partial words are concatenations of the input sequence and two or more characters. In some embodiments, the user selects a partial word to partially complete the input sequence. The device may then identify and present additional character sequences to extend further or complete the input sequence.

[0062] In some embodiments, partial words and complete words are provided to the user as separate choices; for example, by presenting complete words with an appended symbol or an appended space character. In some embodiments, options in addition to the highest-weighted character sequences may be provided to the user. The additional options may include a comprehensive plurality of sequences, each of the plurality of sequences being a concatenation of the input sequence and a distinct character from a predetermined set of supported characters. These sequences need not represent words. Their inclusion would make it possible for the user to enter arbitrary sequences of characters. For example, if the language is English and the input sequence is "the," then the plurality of sequences may include "thea," "theb," "thee," and so forth, up to and including "thez"; followed by "the0," "the1," "the2," and so forth, up to and including "the9"; followed by "the-", "the'," "the!," "the %," "the&," "the=," and various other continuations involving punctuation or common symbols. In some embodiments, the highest-weighted character sequences are presented first in a list, to enable rapid access, and the comprehensive plurality of single-character continuations is presented next.

[0063] It should be appreciated, however, that there may be no candidate sequence that satisfies the usage frequency criteria. In other words, the input sequence node may have no child node remaining. In this situation, operations **318**-**320** may be skipped.

[0064] In some embodiments, identification of the character tree and the nodes having weights that satisfy the usage frequency criteria do not require copying or modifying of the usage frequency tree **146**. Rather, a list of node pointer-weight pairs that represent the current set of children nodes of the input sequence node that satisfy the usage frequency criteria may be stored in the memory **102**. Whenever a grandchild node of the input sequence node is promoted, a pointer to the node and the weight of the node is added to the list, and the weight of the parent node of the promoted grandchild node is updated in the list rather than in the usage frequency tree **146**. Information from the usage frequency tree **146**, such as usage frequency weights, actual character sequences, and descendant nodes, may be accessed via the node pointers. If a child node of the input sequence node no longer satisfies the usage frequency weight criteria, it may be removed from the list. When there are no additional nodes to be added to or removed from the list, the list may be sorted based on the usage frequency weights in accordance with operation **318** described above. Further details are described in relation to FIGS. **7**A-**7**C below.

[0065] Attention is now directed toward FIGS. **5**A-**5**D, which illustrate character sequence trees identified during the process for selecting candidate sequences in accordance with some embodiments. FIG. **5**A shows a character sequence tree **500** (with the root node, corresponding to the empty string, omitted for convenience) where the input sequence node corresponds to an input sequence "c." All descendents of node "c," which includes any character sequence with "c" as a proper prefix, are potential candidate sequences. Children nodes of "c" include nodes corresponding to sequences "ca," "cb," "cc," and so forth, up to and including "cx," "cy," and "cz." Each of these children nodes has subtrees. As described above, children nodes that do not satisfy predefined usage frequency criteria, and their subtrees, are removed (e.g., marked as "removed" without actually removing the nodes from the tree). For example, if nodes "cb," "cc," . . . , "cx," and "cz" do not satisfy the usage frequency criteria, they are removed from the tree **500** as shown in FIG. **5**A, leaving nodes "ca," "cy," and their subtrees.

[0066] As described above in relation to FIGS. **3**A-**3**B, grandchildren nodes of the input sequence node may be promoted to become children nodes of the input sequence nodes.