The function $add(L, c_U, c_I, c_N)$

$add(L, u, i, w)$ {
if $((L, u, w) \notin \mathcal{U}_i$ { add (L, u, w) to $\mathcal{U}_i$, add (L, u, i, w) to $\mathcal{R}$ } }

$add(L, u, i, w)$ takes an algorithm line label, L, a GSS node u, an integer i and an SPPF node w, and creates the descriptor (L, u, i, w), if it does not already exist.

For convenience the set $\mathcal{U}$ is divided into subsets $\mathcal{U}_j$ containing those descriptors whose third entry is j. If $add()$ creates a new descriptor (L, u, i, w) it adds it to $\mathcal{R}$ and $\mathcal{U}_i$.

The role of $\mathcal{U}$ is to prevent descriptors being created repeatedly in ambigous grammars. In particular, it is this that prevents the alogrithm from nontermination on left recursive grammars. (Loops in the GSS ensure that left recursion is correctly handled.)