Currently, the ethereum state data structure is a two-layer trie. At the top there is the account tree, and each account is a 4-part data structure consisting of the nonce, balance, code and storage, where the storage is itself a key/value trie. This model has benefits, but it also has led to considerable implementation complexity. It turns out that we can get most of the benefits of the two-layer model by embedding

it inside a single trie, with objects relating to the same account simply residing beside

each other in the main tree, having the address of the account as a common prefix. That is, now we have:

main_tree[sha3(ADDRESS)] = acct = (nonce, balance, code, storage tree root)
storage_tree[sha3(KEY)] = VALUE

But we can instead have:

main_tree[sha3(ADDRESS) + \x00] = nonce main_tree[sha3(ADDRESS) + \x01] = balance main_tree[sha3(ADDRESS) + \x02] = code main_tree[sha3(ADDRESS) + \x03 + sha3(KEY)] = VALUE

The main benefits of this would be a simple implementation.