

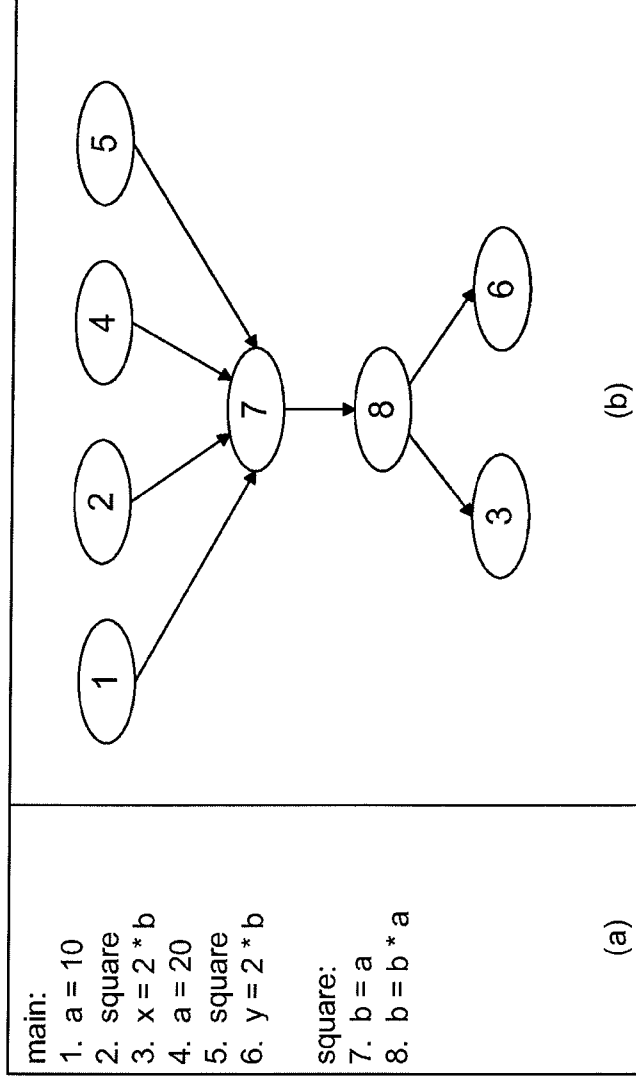


FIG. 1 (prior art)

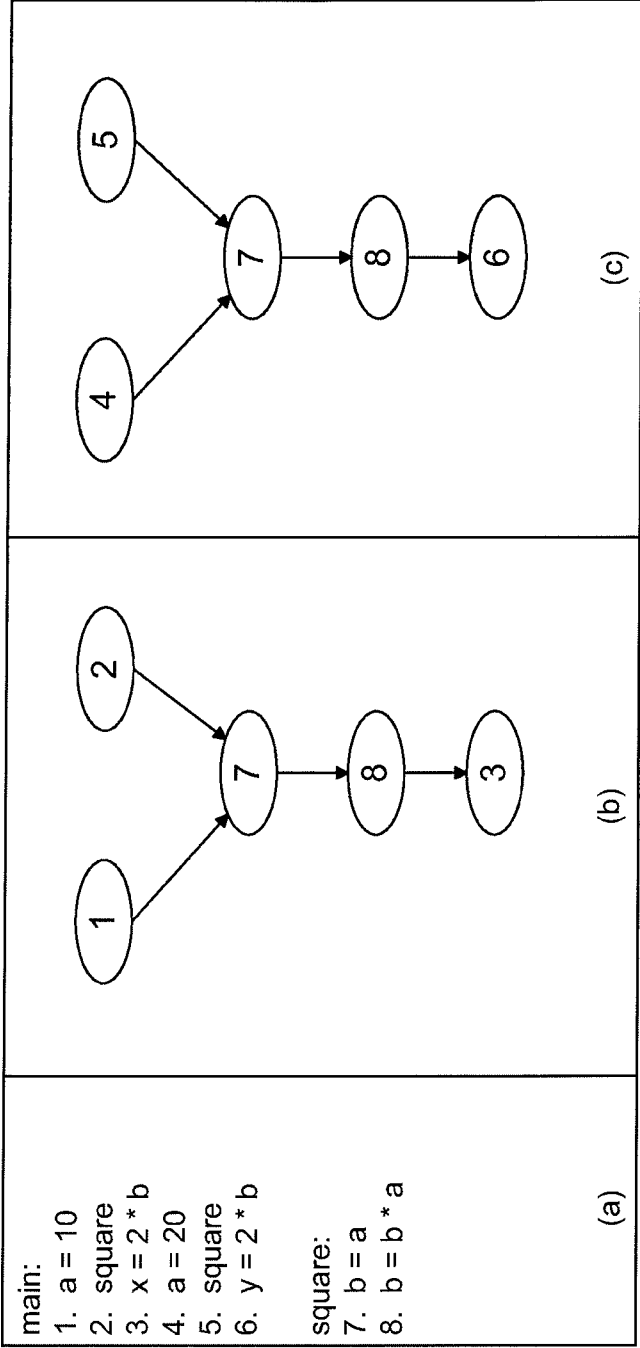


FIG. 2

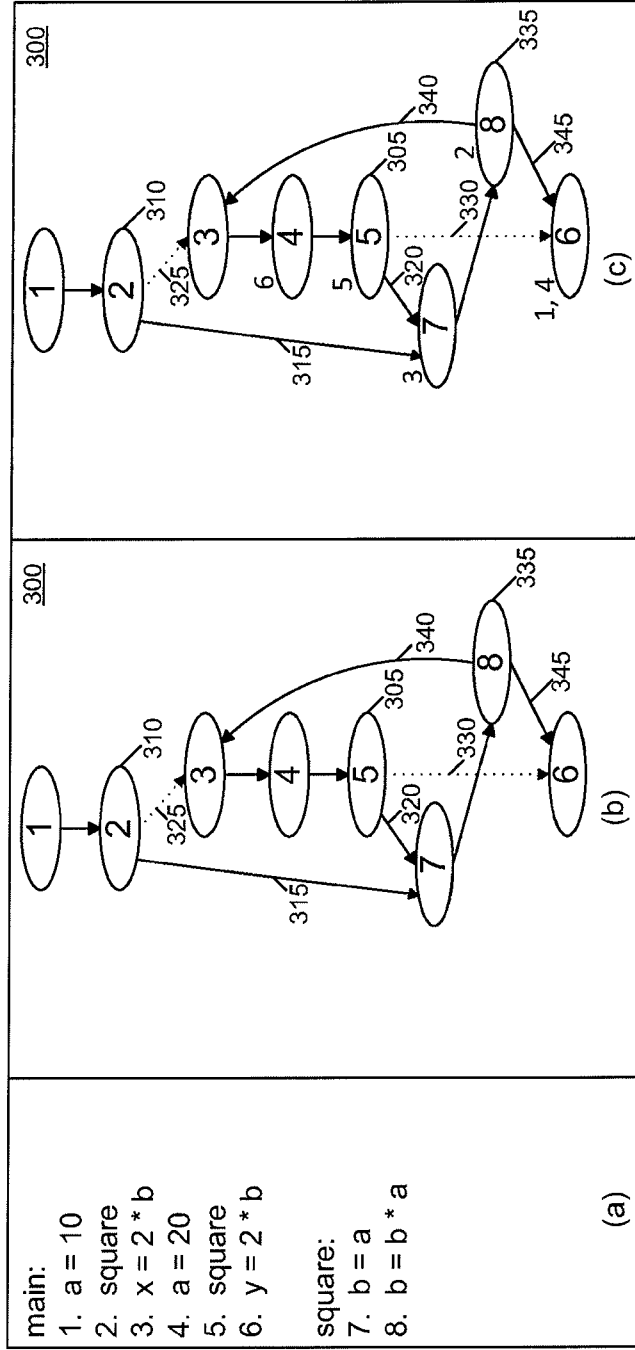


FIG. 3

Function sliceBackwards(*inst*, *vals*, *start*)**inst** :An instruction to start slicing from**vals** :A set of criterion values**start** :A boolean indicating if this call is the start of the slice

1. (flowIn, calls) $\leftarrow$ Subslice(inst, vals, start);
2. foreach (callsite, target) $\in$ calls do
3. callIn $\leftarrow$ flowIn [target];
4. sliceBackwards(callsite, callIn, false);

Function Subslice(*inst*, *vals*, *start*)**inst** :An instruction to start slicing from**vals** :A set of criterion values**start** :A boolean indicating if this call is the start of the slice

- 1 flowIn[inst] $\leftarrow$ vals;
- 2 calls $\leftarrow$ {};
- 3 **if** start is false **then**
- 4 flowIn[inst] $\leftarrow$ flowIn[inst] – use[inst];
- 5 if flowIn[inst] changed **then**
- 6 slice $\leftarrow$ slice $\cup$ {inst};
- 7 worklist $\leftarrow$ pred[inst];
- 8 **while** worklist *not empty* **do**
- 9 remove (node, edgeType, nodeSucc) from worklist;
- 10 oldIn $\leftarrow$ f lowIn[node];
- 11 foreach (succ, succType) $\in$ succ[node] **do**
- 12 if succType is not a call or return **then**
- 13 flowout $\leftarrow$ flowout $\cup$ flowIn[succ];
- 14 **switch** edgeType **do**
- 15 **case** call
- 16 calls $\leftarrow$ $\cup$ {(node, nodeSucc)};
- 17 continue;
- 18 **case** return
- 19 (downFlow, downCalls) $\leftarrow$ Subslice(node, flowIn[nodeSucc], false);
- 20 downFlowList $\leftarrow$ downFlowList + downFlow;
- 21 worklist $\leftarrow$ (preceding[nodeSucc], pseudoEdge) + worklist;
- 22 **case** pseudoEdge
- 23 foreach downIn $\in$ downFlowList **do**
- 24 foreach (succ, succType) $\in$ succ[node] **do**
- 25 if succType is a call **then**
- 26 flowOut $\leftarrow$ flowOut $\cup$ downIn[succ];
- 27 flowIn[node] $\leftarrow$ flowOut;
- 28 **if** (flowOut $\cap$ def[node]) $\neq$ {} **then**
- 29 slice $\leftarrow$ slice $\cup$ {node};
- 30 flowIn[node] $\leftarrow$ (flowIn[node]-def[node]) $\cup$ use[node];
- 31 **if** flowIn[node] $\neq$ oldIn **then**
- 32 worklist $\leftarrow$ worklist + pred[node] + controlDep[node];
- 33 **return** (flowIn, calls);

FIG. 4

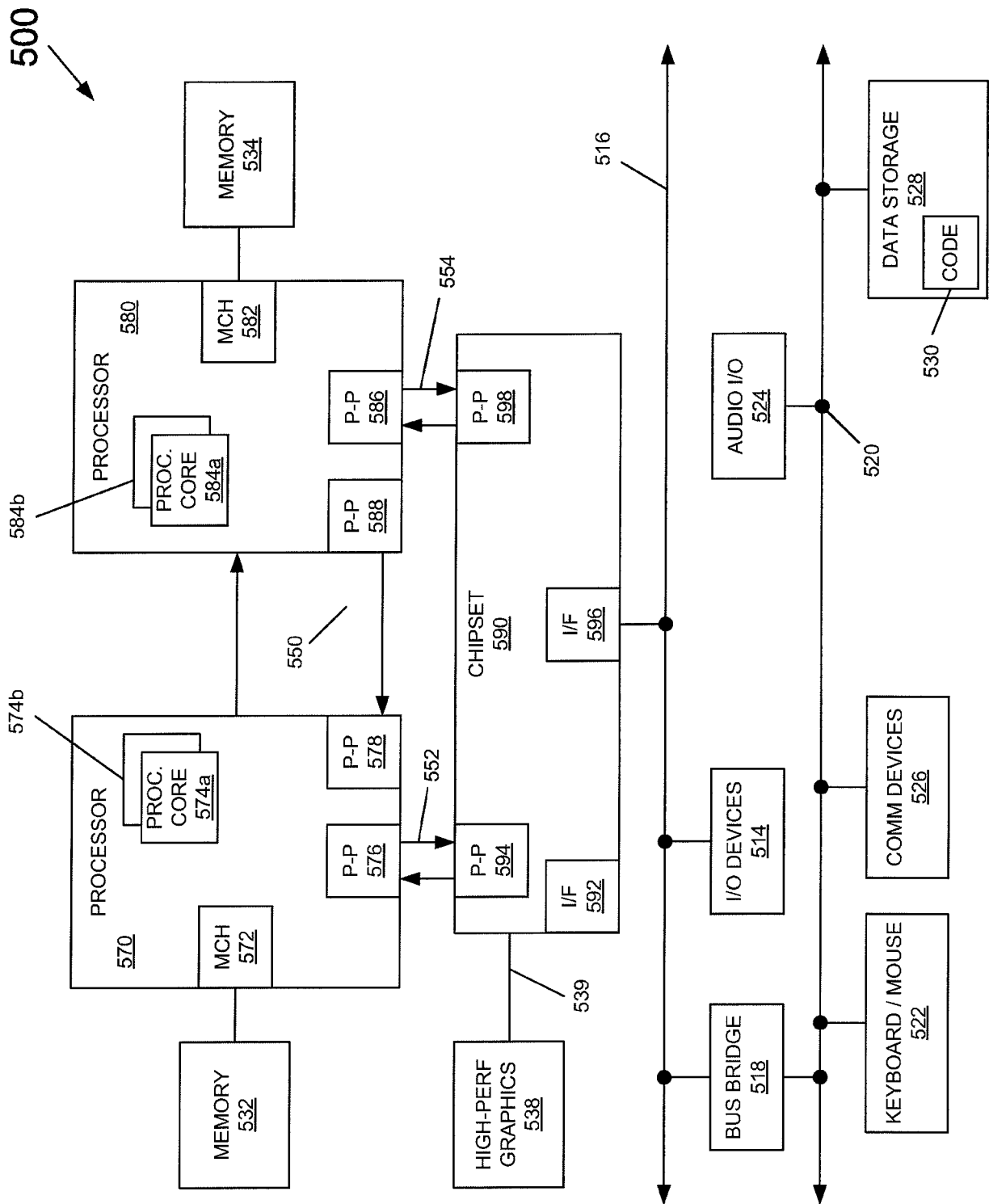


FIG. 5