The solution to the conditional branching problem may be broken into two parts: methods to optimize correct branch selection and methods to prevent errors as a result of conditional branch instructions. Errors are prevented by setting the following guideline: ● While instructions may be executed out of order, they must be completed in the correct order. Since branches and subtle data dependencies can occur, the execution of an instruction out of order may or may not be valid, so the instruction is executed hypothetically, that is, on the assumption that its execution will be useful. ● A few systems place the liability for error prevention on the assembly language programmer or program language compiler by requiring that a certain number of instructions following a conditional branch instruction be independent of the branch. In these systems, one or more instructions sequentially following the branch are always executed, regardless of the outcome of the branch. There are various creative methods that are used in CPUs to optimize conditional branch processing. ● One possible solution to this problem is to maintain two separate instruction fetch pipelines, one for each possible branch outcome. Instructions may be executed speculatively from both branches until the correct pipeline is known. Another solution is to have the CPU attempt to predict the correct path based on program usage or past performance.